

# Package ‘vegdata’

December 1, 2025

**Title** Access Vegetation Databases and Treat Taxonomy

**Version** 1.9.15

**Description** Handling of vegetation data from different sources ( Turboveg 2.0 <<https://www.synbiosys.alterra.nl/turboveg/>>; the German national repository <<https://www.vegetweb.de>> and others. Taxonomic harmonization (given appropriate taxonomic lists, e.g. the Euro+Med list <<https://eurosl.infinitemature.org>>).

**License** GPL (>= 2)

**URL** <https://git.loe.auf.uni-rostock.de/jansen/vegdata.git>

**Depends** foreign, R (>= 3.5.0)

**Imports** curl (>= 2.4), DBI (>= 0.6-1), dbplyr (>= 1.0.0), dplyr (>= 0.7.0), hoardr (>= 0.1.0), data.table, forcats, magrittr (>= 1.5), plyr, RSQLite (>= 1.1.2), stringr, utils, xml2

**Suggests** indicpecies, interp, knitr, labdsv, rmarkdown, uuid, vegan

**VignetteBuilder** knitr

**Encoding** UTF-8

**LazyData** Yes

**RoxygenNote** 7.3.3

**NeedsCompilation** no

**Author** Florian Jansen [aut, cre]

**Maintainer** Florian Jansen <[florian.jansen@uni-rostock.de](mailto:florian.jansen@uni-rostock.de)>

**Repository** CRAN

**Date/Publication** 2025-12-01 17:00:02 UTC

## Contents

vegdata-package . . . . .	2
child . . . . .	3
comb.species . . . . .	4
cwm . . . . .	4

db_download	6
db_path	7
elbaue	7
lc.0	8
lc.1	8
lc.all	9
parent	9
parse.taxa	10
recoding.taxa	10
sql_collect	11
src_vegdata	11
syn	12
syntab	13
tax	14
tax.refl	15
taxlevels	16
taxname.abbrev	16
taxname.removeAuthors	17
taxname.simpl	17
taxval	18
TCS.replace	20
tv.bib	21
tv.coverperc	22
tv.db	22
tv.home	23
tv.metadata	23
tv.obs	24
tv.readXML	25
TV.replace	26
tv.site	26
tv.traits	27
tv.veg	28
tv.write	30
vegdata	31
<b>Index</b>	<b>32</b>

---

vegdata-package	<i>Functions to access data from vegetation databases and evaluate taxon names</i>
-----------------	--

---

## Description

Handling of vegetation data from different sources ( Turboveg 2.0 <https://www.synbiosys.alterra.nl/turboveg/>; the German national repository <https://www.vegetweb.de> and others. Taxonomic harmonization (given appropriate taxonomic lists, e.g. the Euro+Med list <https://euros1.infinitemature.org>).

**Author(s)**

**Maintainer:** Florian Jansen <florian.jansen@uni-rostock.de>

**See Also**

Useful links:

- <https://git.loe.auf.uni-rostock.de/jansen/vegdata.git>

---

child	<i>Search taxonomic reference lists including concept synonymy and taxonomic hierarchy.</i>
-------	---

---

**Description**

Search all (accepted) children of a taxon down to gen generations

**Usage**

```
child(x, refl = tax.refl(), gen = 3, syn = FALSE, include.parent = FALSE, ...)
```

**Arguments**

x	Species number, lettercode or species name(s)
refl	Taxonomic reference list
gen	Number of child generations to return
syn	Should synonyms be included in results
include.parent	Should the parent taxon be included in results
...	additional paarmeters for function tax

**Details**

*concept:* GermanSL is a list with a single taxon view according to the standard lists of the different taxon groups (e.g. Wisskirchen and Haeupler for higher plants, see). Nevertheless a huge number of synonyms is included which allows in many cases the transformation into different concepts. For illustration the concept of *Armeria maritima* from Korneck 1996 is included, which accepts e.g. *Armeria maritima ssp. bottendorfensis*. *parse.taxa:* parse genus and epitheta from name strings. *taxname.removeAuthors* Remove name authors from full scientific name strings.

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>

**References**

Jansen, F. and Dengler, J. (2008) GermanSL - eine universelle taxonomische Referenzliste für Vegetationsdatenbanken. *Tuexenia*, 28, 239-253.

---

comb.species	<i>Combine species in vegetation matrix</i>
--------------	---

---

### Description

Combine species in vegetation matrix

### Usage

```
comb.species(x, sel, newname, refl)
```

### Arguments

x	(veg) vegetation matrix of class veg
sel	(character) vector of species to combine
newname	(character) name of the new taxon
refl	(character) Name of taxonomic reference list

---

cwm	<i>Indicate site conditions with community weighted mean values of traits or with mode of gradient classes (sum of species amplitudes).</i>
-----	---

---

### Description

Calculates community weighted mean trait values, like mean Ellenberg indicator values. Alternatively (method = 'mode') environmental conditions can be calculated according to the concept of sums of amplitudes of species along ecological gradients.

### Usage

```
cwm(
  veg,
  refl,
  trait.db = "ecodbase.dbf",
  ivname,
  keyname = "LETTERCODE",
  method = c("mean", "mode"),
  weight,
  db,
  ...
)
```

**Arguments**

veg	Vegetation matrix with plots in rows and species in columns
refl	Name of Turboveg taxonomic reference list
trait.db	data frame with species trait values
ivname	Name of the trait in trait.db to be used
keyname	Name of the column in trait dataframe to join with colnames of veg table
method	mean (weighted value of single traits, or mode (maximum) of trait classes)
weight	additional weight, e.g niche breath of species
db	name of Turboveg database
...	additional arguments

**Details**

Trait values of 0 will be handled as NA values because Turboveg dBase can not handle NA values properly.

**Value**

Vector with the ecological classification of sites. Either mean trait values or mode of gradient classes.

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>

**Examples**

```
## Not run:
db <- 'elbaue'
veg <- tv.veg(db, cover.transform='sqrt', check.critical = FALSE)
site <- tv.site(db, verbose = FALSE)

#' Mean indicator values of Ellenberg F values
mEIV_F <- cwm(veg, ivname = 'OEK_F', method = 'mean')
plot(site$MGL, mEIV_F, xlab = 'Mean groundwater level')

## End(Not run)
```

---

`db_download`*Download taxonomic databases*

---

**Description**

Download taxonomic databases

**Usage**

```
db_download_eurosl(version = "latest", verbose = TRUE, overwrite = FALSE)
```

```
db_download_germansl(version = "latest", verbose = TRUE, overwrite = FALSE)
```

**Arguments**

<code>version</code>	(character) desired version number of the list
<code>verbose</code>	(logical) Print messages. Default: 'TRUE'
<code>overwrite</code>	(logical) If 'TRUE' force an update by overwriting previously downloaded data. Default: 'FALSE'

**Details**

Downloads sql database, cleans up unneeded files, returns path to sql file

**Value**

(character) path to the downloaded SQL database

**See Also**

[.my\_cache]

**Examples**

```
## Not run:  
# EuroSL  
# db_download_eurosl()  
# src_eurosl()  
  
# GermanSL  
# db_download_germansl()  
# db_download_germansl(overwrite=TRUE) # overwrite - download again  
# src_germansl()  
  
## End(Not run)
```

---

db_path	<i>database path</i>
---------	----------------------

---

### **Description**

database path

### **Usage**

db\_path(db)

### **Arguments**

db (character) db name. one of: eurosl, germansl

---

elbaue	<i>This is an example vegetation dataset to be included in package vegdata. Use tv.veg('elbaue') and tv.site('elbaue') to load.</i>
--------	---

---

### **Description**

This is an example vegetation dataset to be included in package vegdata. Use tv.veg('elbaue') and tv.site('elbaue') to load.

### **Author(s)**

Florian Jansen

### **Source**

<https://www.vegetweb.de>

---

lc.0

*Layer combinations*

---

### **Description**

Example tables with layer codes and how they should be combined in vegetation analyses. lc.0: do not combine any layers

### **Usage**

lc.0

### **Format**

A data frame with rows and 2 variables:

**LAYER** Layer code, i.e. 0:9 for Turboveg

**COMB** Combinations. Same integer means, they will be combined

---

lc.1

*Layer combinations*

---

### **Description**

datasets with layer codes and how they should be combined in vegetation analyses.

### **Usage**

lc.1

### **Format**

A data frame with rows and 3 variables:

**LAYER** Layer code, i.e. 0:9 for Turboveg

**COMB** Combinations. Same integer means, they will be combined



---

lc.all	<i>Layer combinations</i>
--------	---------------------------

---

**Description**

A data.frame to combine all layers in a vegetation plot

**Usage**

```
lc.all
```

**Format**

A data frame with rows and 2 variables:

**LAYER** Layer code, i.e. 0:9 for Turboveg

**COMB** Combinations. Same integer means, they will be combined

---

parent	<i>#' Parents of a taxon</i>
--------	------------------------------

---

**Description**

*#' Parents of a taxon*

**Usage**

```
parent(x, refl = tax.refl(), rank, taxlevels, ...)
```

**Arguments**

x	Species number, lettercode or species name(s)
refl	Taxonomic reference list
rank	taxonomic level of taxa to find
taxlevels	A data frame defining the hierarchy of taxon levels, see vegdata::taxlevels
...	additional attributes for function tax

**Details**

*concept*: GermanSL is a list with a single taxon view according to the standard lists of the different taxon groups (e.g. Wisskirchen and Haeupler for higher plants, see). Nevertheless a huge number of synonyms is included which allows in many cases the transformation into different concepts. For illustration the concept of *Armeria maritima* from Korneck 1996 is included, which accepts e.g. *Armeria maritima* ssp. *bottendorffensis*. *parse.taxa*: parse genus and epitheta from name strings. *taxname.removeAuthors* Remove name authors from full scientific name strings.

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>

**References**

Jansen, F. and Dengler, J. (2008) GermanSL - eine universelle taxonomische Referenzliste für Vegetationsdatenbanken. *Tuexenia*, 28, 239-253.

---

parse.taxa	<i>Parse taxon strings into genus part and epitheta</i>
------------	---

---

**Description**

Parse taxon strings into genus part and epitheta

**Usage**

```
parse.taxa(x, epis)
```

**Arguments**

x	(character) taxon names
epis	(character) vector of separators for epithets (like e.g. "subsp.")

---

recoding.taxa	<i>Recode taxon names, lettercodes or ID's</i>
---------------	--

---

**Description**

Recode taxon names, lettercodes or ID's

**Usage**

```
recoding.taxa(x, names = c("shortletters", "Numbers", "ScientificNames"), refl)
```

**Arguments**

x	vector of species
names	one of 7digit shortletter codes, species id's or scientific species names
refl	(character) name of taxon reference list

---

sql_collect	<i>Query and get data back into a data.frame</i>
-------------	--

---

**Description**

Query and get data back into a data.frame

**Usage**

```
sql_collect(src, query, ...)
```

**Arguments**

src	(src) An 'src' object, result of calling [src_germansl()], [src_eurosl()]
query	(character) A SQL query
...	further args passed on to [dplyr::tbl()]

**Details**

we run [dplyr::tbl()], then [dplyr::collect()]

**Examples**

```
## Not run:
src <- src_germansl()
sql_collect(src, "select * from GermanSL limit 5")
## or pipe the src to sql_collect
src %>% sql_collect("select * from GermanSL limit 5")

## End(Not run)
```

---

src_vegdata	<i>src - dplyr src objects</i>
-------------	--------------------------------

---

**Description**

src - dplyr src objects

**Usage**

```
src_eurosl(path = db_path("eurosl"), ...)

src_germansl(path = db_path("germansl"), ...)
```

**Arguments**

path (character) path to SQLite database. by default we use the function [db\_path()] to get the path

... Further args passed on to [DBI::dbConnect()]

**Value**

an src object

**Examples**

```
## Not run:
# src_eurosl()
# src_germansl()

## End(Not run)
```

---

syn

*Search synonyms of a taxon*

---

**Description**

Search synonyms of a taxon

**Usage**

```
syn(x, refl = tax.refl(), ...)
```

**Arguments**

x Species number, lettercode or species name(s)

refl Taxonomic reference list

... additional attributes for function tax

**Details**

*concept*: GermanSL is a list with a single taxon view according to the standard lists of the different taxon groups (e.g. Wisskirchen and Haeupler for higher plants, see). Nevertheless a huge number of synonyms is included which allows in many cases the transformation into different concepts. For illustration the concept of *Armeria maritima* from Korneck 1996 is included, which accepts e.g. *Armeria maritima ssp. bottendorfensis*. *parse.taxa*: parse genus and epitheta from name strings. *taxname.removeAuthors* Remove name authors from full scientific name strings.

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>

## References

Jansen, F. and Dengler, J. (2008) GermanSL - eine universelle taxonomische Referenzliste für Vegetationsdatenbanken. *Tuexenia*, 28, 239-253.

---

syntab	<i>Syntaxonomic frequency tables</i>
--------	--------------------------------------

---

## Description

Calculate and display relative or absolute frequency tables with or without use of function `multipatt` from package `indicpecies`

## Usage

```
syntab(
  veg,
  clust,
  type = c("rel", "abs", "mean.cover"),
  mupa,
  dec = 0,
  refl,
  ...
)
```

```
## S3 method for class 'syntab'
print(x, zero.print = ".", trait, limit = 1, minstat = 0, alpha = 0.05, ...)
```

## Arguments

<code>veg</code>	Vegetation dataframe
<code>clust</code>	Vector with cluster information with length equal to number of rows of <code>veg</code>
<code>type</code>	Relative (species or type) or absolute frequency, mean species response values or strength of association.
<code>mupa</code>	Either logical for (not) using <code>multipatt</code> from package <code>indicpecies</code> to detect significance of cluster association strength or supply output from previous use of <code>multipatt</code> .
<code>dec</code>	Number of decimals in result.
<code>refl</code>	Name of <code>Turboveg</code> taxonomic reference list to use for fullnames.
<code>...</code>	additional arguments
<code>x</code>	Object from function <code>syntab</code>
<code>zero.print</code>	Replacement for zero values.
<code>trait</code>	Optional vector of trait values to be plotted behind the species.
<code>limit</code>	Minimum value to display.
<code>minstat</code>	Minimal indicator value
<code>alpha</code>	Significance threshold.

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>

**See Also**

Package `indicspecies` with function `multipatt` for indicator species analysis along multiple cluster combinations

**Examples**

```
## Not run:
elbaue <- tv.veg('elbaue')
elbaue.env <- tv.site('elbaue')
clust <- vector('integer', nrow(elbaue.env))
clust[elbaue.env$MGL < -50 & elbaue.env$SDGL < 50] <- 1
clust[elbaue.env$MGL < -50 & elbaue.env$SDGL >= 50] <- 2
clust[elbaue.env$MGL >= -50 & elbaue.env$SDGL >= 50] <- 3
clust[elbaue.env$MGL >= -50 & elbaue.env$SDGL < 50] <- 4
levels(clust) <- c('dry.ld', 'dry.hd', 'wet.hd', 'wet.ld')
st <- syntab(elbaue, clust, mupa=TRUE)
traits <- tv.traits()
m <- match(rownames(st$syntab), traits$LETTERCODE, nomatch = 0)
trait <- traits[m, c("OEK_F", "OEK_N")]
rownames(trait) <- traits$LETTERCODE[m]
print(st, limit=30, trait=trait)
#' Configure the syntaxonomic table
sttable <- st$syntab
sttable <- sttable[sttable$p.value < 0.05 & !is.na(sttable$p.value),
!names(sttable) %in% c('stat')]
taxa <- tax(rownames(sttable))
rownames(sttable) <- taxa[match(rownames(sttable), taxa$LETTERCODE, nomatch = 0), 'TaxonName']
write.csv(sttable, 'sttable.csv')

## End(Not run)
```

---

tax

*Search taxonomic reference lists including concept synonymy and taxonomic hierarchy.*

---

**Description**

Input is either species number (integer), shortletter (7 characters) or full (exact!) species name.

**Usage**

```
tax(x, refl = tax.refl(), syn = TRUE, concept = NULL, strict = FALSE, idtype)
```

**Arguments**

x	Species number, lettercode or species name(s)
refl	Taxonomic reference list
syn	Return also synonym names
concept	Name of the file with an alternative taxon view stored in the reference list directory, see details.
strict	Exact match or partial matching with <a href="#">grep</a>
idtype	is x of type 'is_id', 'is_shortletter', or 'is_scientific'

**Details**

*concept*: GermanSL is a list with a single taxon view according to the standard lists of the different taxon groups (e.g. Wisskirchen and Haeupler for higher plants, see). Nevertheless a huge number of synonyms is included which allows in many cases the transformation into different concepts. For illustration the concept of *Armeria maritima* from Korneck 1996 is included, which accepts e.g. *Armeria maritima* ssp. *bottendorffensis*. *parse.taxa*: parse genus and epitheta from name strings. *taxname.removeAuthors* Remove name authors from full scientific name strings.

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>

**References**

Jansen, F. and Dengler, J. (2008) GermanSL - eine universelle taxonomische Referenzliste für Vegetationsdatenbanken. *Tuexenia*, 28, 239-253.

---

tax.refl	<i>Taxon reference list to be used</i>
----------	--

---

**Description**

Taxon reference list to be used

**Usage**

```
tax.refl(refl, db, tv_home)
```

**Arguments**

refl	name of taxonomic reference list
db	optional: Turboveg database name
tv_home	Turboveg installation path

---

taxlevels	<i>Taxonomic hierarchy levels</i>
-----------	-----------------------------------

---

**Description**

hierarchy of taxon levels

**Usage**

taxlevels

**Format**

A data frame with rows and 4 variables:

**rank** Layer code, i.e. 0:9 for Turboveg

**level** Combinations. Same integer means, they will be combined

**Euro.Med** taxon level name in <http://www.europusmed.org/> database

**description** Explanation of level codes

---

taxname.abbr	<i>Standardisation of taxonomic names, especially taxon rank indicators and hybrid signs name taxname.abbr</i>
--------------	--

---

**Description**

Standardisation of taxonomic names, especially taxon rank indicators and hybrid signs name taxname.abbr

**Usage**

taxname.abbr(x, hybrid = TRUE, concept = FALSE, cf = TRUE, ...)

**Arguments**

x	(integer or character) Species number, lettercode or species name(s)
hybrid	(logical) remove hybrid markers for comparisons
concept	(logical) remove concept additions like "s. str.", "s. l.
cf	(logical) remove 'in doubt' marker
...	additional attributes

**Author(s)**

Florian Jansen [florian.jansen@uni-rostock.de](mailto:florian.jansen@uni-rostock.de)



---

taxname.removeAuthors *Remove name authors from taxon names*

---

**Description**

Remove name authors from taxon names

Extract name authors from taxon names

**Usage**

```
taxname.removeAuthors(x, rankattr)
```

```
taxname.authors(x, rankattr)
```

**Arguments**

x (character) vector of taxon names

rankattr (character) vector of rank attributes used in the taxonname strings

---

taxname.simpl *Simplify name parts for better string matching*

---

**Description**

Simplify name parts for better string matching

**Usage**

```
taxname.simplify(  
  x,  
  genus = TRUE,  
  epithet = TRUE,  
  concept = TRUE,  
  rank = TRUE,  
  tax.status = TRUE,  
  ...  
)
```

**Arguments**

x (integer or character) Species number, lettercode or species name(s)

genus (logical) simplify genus name part

epithet (logical) simplify epithet(s)

concept	(logical) remove name parts which describe taxon concept size like "s. str.", "s. l."
rank	(logical) remove rank specifications
tax.status	(logical) remove taxon status like 'nom. illeg.' or 'auct.'
...	additional attributes

### Details

taxname.abbr will be applied beforehand automatically. The function simplifies name parts which are empirically unstable, i.e. *sylvatica* might also written as *silvatica*, or *majus* vs. *maius*. Sex of latin genus or epithet name parts often change and are therefore deleted (us vs. a, ea vs. eos, etc.). Hybrid signs are removed. taxname.simpl works well for plant names, but be careful with very long name lists or if combined with animal taxa which are sometimes very short and can be confused after applying taxname.simpl

### Author(s)

Florian Jansen [florian.jansen@uni-rostock.de](mailto:florian.jansen@uni-rostock.de)

---

taxval

*Handling of taxonomy in vegetation data.*

---

### Description

Performs taxonomic valuation of species names according to synonymy, taxonomic level, unambiguous biotic content etc. Necessary prerequisite is information about taxonomic status (synonymy) and hierarchy (next higher aggregate). Until now only applicable for reference list 'GermanSL' (>= version 1.1, see References Section), which is valid in Germany and adjacent countries.

### Usage

```
taxval(
  obs,
  refl,
  db,
  ag = c("conflict", "adapt", "preserve"),
  rank,
  mono = c("species", "higher", "lower", "preserve"),
  monolist = "monotypic-D",
  maxtaxlevel = "AGG",
  ranklevels = NULL,
  check.critical = TRUE,
  interactive = FALSE,
  ...
)
```

## Arguments

obs	data.frame of observations in TURBOVEG format, for example loaded with <a href="#">tv.obs</a>
refl	Name of taxonomic reference list
db	a name of a Turboveg database directory containing tvabund.dbf, tvhabita.dbf and twin.set
ag	Treatment of children and parents within the dataset, see details
rank	If ag='adapt', rank specifies the taxonomic rank to which taxa should be coarsened to. All higher taxa in this taxonomic tree will be deleted, see maxtaxlevel.
mono	Should monotypic taxa be combined at subspecies = 'lower' or species level = 'higher'
monolist	Name of monotypic species list, must be in dBase format and in the same directory as the reference list, e.g. "monotypic-D" for the area of Germany.
maxtaxlevel	Maximum taxonomic levels to be used. See details.
ranklevels	Ordered factor of taxonomic ranks, see vegdata:taxlevels
check.critical	Check for critical names in your dataset and give warnings.'
interactive	Do you want to adapt the list of changes.
...	Other parameters passed to functions.

## Details

Working with vegetation datasets, especially from different sources needs taxonomic valuation. The function tries to automate this process. Therefore the German taxonomic reference list (GermanSL, <https://germansl.infinitemature.org>) contains additional taxon attributes (tax.dbf) and monotypic taxa of Germany (monotypic.dbf). Without an appropriate species list (see [tax](#)) the function will not work.

The taxonomic reference list needs Taxonrank corresponding to values given in taxlevels Possible values for adapting the taxonomic hierarchy within the dataset (child/parent taxa) are: preserve: Leave everything untouched. conflict: Dissolve only in case of conflicts, e.g. if a subspecies occurs also at the species level within the same dataset. In this case the subspecies will be aggregated to the higher level. adapt: Dissolve all nested taxa to e.g. species level for option ag. For this option also option rank, specifying the rank to which the taxa shall be adapted, must be given.

Monotypic taxa, e.g. a species which occur only with 1 subspecies in the survey area. They have to be combined, since otherwise two different (valid) taxa would denominate the same entity. If lower the higher taxon (e.g. species rank) is replaced by the lower level (subspecies rank). If neither lower nor higher monotypic species are preserved. Since the list of monotypic species strongly depends on the considered area you have to choose, which area is covered by your database and create an appropriate list of monotypic taxa. Within the package "monotypic-D.csv" is provided as a compilation of monotypic species within the GermanSL list.

Option maxtaxlevel determines the maximum taxonomic level within the given names, which should be used. All higher taxon observations are deleted. If you have a single field observation determined as *Asteraceae spec.* all your observations of taxa from that family will be aggregated to the family level, if you choose ag=conflict.

**Interactive** If you want to manually adapt the taxonomic harmonization `interactive=TRUE` will create a table with all original names and `NewTaxonID`'s according to the chosen rules. The table will be saved as `taxvalDecisionTable.csv` in your actual working directory. You can manipulate the column `NewTaxonID`. If you run `taxval` again (e.g. through function `tv.veg`) and a file with this name exist in your working directory, it will be used.

### Value

Functions return the input dataframe of observations with harmonised taxon numbers.

### Author(s)

Florian Jansen <florian.jansen@uni-rostock.de>

### References

Jansen, F. and Dengler, J. (2008) GermanSL - eine universelle taxonomische Referenzliste für Vegetationsdatenbanken. *Tuexenia*, 28, 239-253. Jansen, F. and Dengler, J. (2010) Plant names in vegetation databases - a neglected source of bias. *Journal of Vegetation Science*, 21, 1179-1186.

### See Also

`tv.veg`, `tv.obs`

### Examples

```
## Not run:
# Turboveg installation needed
obs <- taxval(db='taxatest')
# For explanations see vignette('vegdata').

veg <- tv.veg('taxatest')
veg <- comb.species(veg, c('ARMEM-E', 'ARMEM-H'))

## End(Not run)
```

---

TCS.replace

*Standardise taxon list field names to match the Taxonomic Concept Transfer Schema (TCS)*

---

### Description

Applies Taxonomic Concept Transfer Schema (TCS) to the different name list conventions of different sources

### Usage

`TCS.replace(x)`

**Arguments**

x (character) string of column names used in data.frames storing taxon lists

**Author(s)**

Florian Jansen [florian.jansen@uni-rostock.de](mailto:florian.jansen@uni-rostock.de)

**References**

Taxonomic Names and Concepts interest group. 2006. Taxonomic Concept Transfer Schema (TCS), version 1.01. Biodiversity Information Standards (TDWG) <http://www.tdwg.org/standards/117>

---

tv.bib	<i>Check bibliographic references from Turboveg codes</i>
--------	---

---

**Description**

Check bibliographic references from Turboveg codes

**Usage**

```
tv.bib(x = "all", db, dict = tv.dict(db), tv_home, ...)
```

**Arguments**

x (character) Turboveg reference code(s), e.g. "000001"

db (character) Database name. Needed to select appropriate TV Dictionary folder.

dict (character) Name of Turboveg Dictionary (term lists for header data) if not the default one.

tv\_home (character) Turbowin installation path. If not specified function [tv.home()] tries to discover.

... additional arguments

**Value**

Dataframe of (selected) bibliographic references (when assigned to an object).

**Author(s)**

Florian Jansen <[florian.jansen@uni-rostock.de](mailto:florian.jansen@uni-rostock.de)>

---

tv.coverperc	<i>Cover code translation</i>
--------------	-------------------------------

---

**Description**

Translate cover code into percentage cover values for Turboveg database observations.

**Usage**

```
tv.coverperc(db, obs, RelScale, tv_home, tvscale, ...)
```

**Arguments**

db	the name of the Turboveg database
obs	dataframe of observations, containing Cover Codes, coded in tvscale.dbf of Turboveg installation
RelScale	Dataframe of CoverScale codes per releve, if empty it is read from the database
tv_home	Path to Turboveg installation
tvscale	Cover scale
...	Further options

**Value**

data.frame of observations with additional column COVER\_PERC

---

tv.db	<i>Read list of available Turboveg2 databases in given Turboveg directory</i>
-------	---

---

**Description**

Read list of available Turboveg2 databases in given Turboveg directory

Read list of available Turboveg2 databases in given Turboveg directory

**Usage**

```
tv.db(path)
```

```
tv.dict(db, tv_home)
```

**Arguments**

path	(character) directory path inside Turboveg/data directory
db	(character) name of Turboveg database/directory
tv_home	(character) path of Turboveg instalation

**Value**

List of databases below specified path

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>

---

tv.home	<i>Where is your Turboveg 2 installation path or the directory with Species, Data, and Popup you want to use?</i>
---------	---

---

**Description**

Reads and sets invisibly option('tv\_home')

**Usage**

```
tv.home(check = FALSE)
```

**Arguments**

check (logical) reset even if option('tv\_home') is already set

**Value**

Reads and sets invisibly option('tv\_home')

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>

---

tv.metadata	<i>Show metainfo of vegetation database or ecodbase</i>
-------------	---

---

**Description**

Showing "metadata.txt" when specified and saved in Turboveg database directory. When db = 'eco' and refl specified, metainfo of species attribute table is displayed.

**Usage**

```
tv.metadata(db, refl, tv_home, filename = 'metadata.txt', ...)
```

**Arguments**

db	Turboveg database name
refl	Turboveg taxonomic reference list, declaration only necessary for ecodbase info
tv_home	Turboveg installation path
filename	Name of metainfo file residing in database directory
...	additional arguments

**Details**

Because Turboveg provides no formalised method to store information about database fields, I suggest to save a simple text file, named for example "metadata.txt" into the directory of your Turboveg database.

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>

---

tv.obs

*Dataframe of plot-species observations directly from Turboveg*

---

**Description**

Dataframe of plot-species observations directly from Turboveg.

**Usage**

```
tv.obs(db, tv_home, ...)
```

**Arguments**

db	(character) Name of your Turboveg database. This is the directory name containing tvabund.dbf, tvhabita.dbf and tvwin.set. Please include pathnames below but not above Turbowin/Data.
tv_home	(character) Turbowin installation path. If not specified function <a href="#">tv.home</a> tries to discover.
...	additional arguments

**Value**

Data.frame of species occurrences in Turboveg format, that is every occurrence is a row with releve number, species number, layer, cover code and optional additional species-plot information.

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>



**See Also**[tv.veg](#)**Examples**

```
## Not run:  
# Turboveg installation needed  
obs <- tv.obs('taxatest')  
head(obs)  
  
## End(Not run)
```

---

`tv.readXML`*Reads a Turboveg XML file*

---

**Description**

Reads Turboveg XML formatted files species-plot observations and site information into a list

**Usage**

```
tv.readXML(file)
```

**Arguments**

`file` (character) Path name of the Turboveg XML file

**Value**

S3 list with elements `tvwin`, `tvadmin`, `site`, and `obs`

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>

**See Also**[tv.veg](#), [tv.site](#)

---

TV.replace	<i>Rename data.frame columns to match Turboveg 2 conventions</i>
------------	--

---

**Description**

Rename data.frame columns to match Turboveg 2 conventions

**Usage**

```
TV.replace(x)
```

**Arguments**

x (character) string vector of column names

---

tv.site	<i>Load site data from Turboveg Database</i>
---------	--

---

**Description**

Loading Turboveg header data and do basic data evaluation. Empty columns are eliminated and warnings about possibly wrong '0' values are performed

**Usage**

```
tv.site(
  db,
  tv_home,
  drop = TRUE,
  common.only = FALSE,
  check.relevenumbers = TRUE,
  replace.names,
  ...
)
```

**Arguments**

db (character) Name of your Turboveg database(s). Directory name containing tv-abund.dbf, tvhabita.dbf and tvwin.set.

tv\_home (character) Turbowin installation path. Optional, if Turbowin is either on "C:/Turbowin" or "C:/Programme/Turbowin".

drop (logical) Drop variables without values.

common.only (logical) Import only header data with the same name in all databases.

check.relevenumbers (logical) Are duplicate PlotObservationID's allowed. Default is FALSE.

replace.names (data.frame) replace variable names. Useful if using multiple source databases. Data frame with names to be replaced in first and replacing names in second column.

... Additional options like dec for type.convert

### Details

Please specify pathnames below but not above Turbowin/Data. Can be a single database or a character vector of multiple databases. In the latter case you have to assure, that all databases use the same taxonomic reference list. You can use the example in the final output line to make a summary statistic for attributes with potentially misleading '0' values. Just delete the \" at beginning and end.

### Value

data.frame of site variables.

### Author(s)

Florian Jansen <florian.jansen@uni-rostock.de>

---

tv.traits	<i>Load species traits from Turboveg reference list</i>
-----------	---

---

### Description

Loading Turboveg ecodbase or any other specified dBase file in this directory and do basic data evaluation. Empty columns are eliminated.

### Usage

```
tv.traits(db, trait.db = "ecodbase.dbf", refl, ...)
```

### Arguments

db	Path name to the Turboveg database directory
trait.db	Name of species trait dBase file, default is 'ecodbase'
refl	Name of the taxonomic reference list, if veg is not loaded with tv.veg
...	additional arguments for tv.traits

### Details

You can use the final output line to make a summary statistic for attributes with potentially misleading '0' values.

### Value

data.frame of ecological traits, see metainfo(refl, eco=TRUE)

**Author(s)**

Florian Jansen <florian.jansen@uni-rostock.de>

---

tv.veg

*Tabulates vegetation tables from Turboveg database*

---

**Description**

Tabulates vegetation tables from Turboveg resp. VegetWeb database, including taxonomic emendation and layer combination. Using various default parameters for the included functions. It is a wrapper for `*tv.obs*`, `*taxval*`, `*tv.coverperc*` and creating a vegetation matrix

**Usage**

```
tv.veg(
  db,
  taxval = TRUE,
  convcode = TRUE,
  lc = c("layer", "mean", "max", "sum", "first"),
  pseudo,
  values = "COVER_PERC",
  spcnames = c("shortletters", "ScientificNames", "Numbers"),
  dec = 0,
  cover.transform = c("no", "pa", "sqrt"),
  obs,
  site,
  refl = tax.refl(),
  RelScale,
  tv_home,
  ...
)
```

**Arguments**

db	Name of your Turboveg database. Directory name containing tvabund.dbf, tvhabita.dbf and tvwin.set. Please specify pathnames below (if you sorted your databases in subfolders) but not above Turbowin/Data.
taxval	Should taxonomic valuation (see <a href="#">taxval</a> ) be performed?
convcode	Should cover code be converted to percentage values?
lc	Layer combination type. Possible values: layer (default), sum, mean or max, see details
pseudo	List used for layer combinations, see details
values	Name of the variable which should be used for the vegetations matrix.
spcnames	Should species numbers be replaced by shortletters or ScientificNames? Layer information is appended with dot.

dec	Number of decimals for cover values in the resulting vegetation matrix.
cover.transform	If you want to transform the abundance values within your samples you can choose 'pa' for presence-absence or 'sqrt' for the dec rounded square root.
obs	Observations, optional
site	plot header data, see <a href="#">tv.site</a>
refl	Taxonomic reference list, optional
RelScale	Vector with Cover Scale code per Releve.
tv_home	Turbovin installation path.
...	additional arguments for included functions

### Details

layer means, the different layers are combined assuming their independence (a species occurring in two layers with a cover of 50% will result in an overall cover of 75%). `sum` will sum up cover values of all layers. With option `pseudo` you can decide, which layers should be combined. Give a list with a combination `data.frame` and second the name of the column for combination. The default is `pseudo = list(lc.1, c('LAYER'))`, where `lc.1` is a `data.frame` `data(lc.1)`, which will combine all tree layers, all shrub layers and all layers below shrubs. An alternative would be `data(lc.all)`, combining all layers. With option `pseudo=NULL` there will be no layer aggregation.

### Value

an object of class `matrix` with (combined) cover values.

### Author(s)

Florian Jansen <[florian.jansen@uni-rostock.de](mailto:florian.jansen@uni-rostock.de)>

### See Also

[taxval](#), [tv.coverperc](#), [tv.obs](#), [tv.site](#)

### Examples

```
## Not run:
vignette("vegdata")
#' If you have Turboveg installed on your computer try for a beginning
#' tv.veg('databasename', tax=FALSE).
args(tv.veg)
help('taxval')

veg <- tv.veg('taxatest')
names(veg)
tv.veg('taxatest', uncertain=list('DET_CERT', data.frame(0:2,c('pres','agg','agg'))),
      pseudo=list(lc.0,'LAYER'), genus = 'delete')

## End(Not run)
```

---

tv.write	<i>Write species-plot observations and site information to Turboveg database.</i>
----------	---

---

### Description

Write species-plot observations and site information to Turboveg database.

### Arguments

x	(data.frame) Either observations data.frame with PlotObservationID, TaxonUsageID and COVER_CODE (see <a href="#">tv.obs</a> ) columns or vegetation matrix of class "veg".
site	(character) Header data for plots.
name	(character) Name of the new database.
tvadmin	(data.frame) Dataframe with plot UUID's and Turboveg columns from TvAdmin.dbf. A new file with new unique identifiers will be created if omitted.
remarks	(data.frame) Remarks in Turboveg format if the comments for individual plots exceed 254 characters. See remarks.dbf in Turboveg databases. An empty file will be created if omitted.
dict	(character) Turboveg dictionary name
cover	(logical) Use of covercodes or (mean) cover percentages, see Details.
drop	(logical) Drop columns which are empty or contain only NA values.
obl	(logical) Add obligatory fields defined in the TV dictionary but not present in the site data table.
overwrite	(logical) Should an existing database be overwritten.
...	additional arguments

### Details

By default Covercode is written to Turboveg. This is only meaningful, if correct CoverScales are given in the site dataframe. Unique plot ID's are stored in \*TvAdmin.dbf\*. If you want to preserve already given UUID's you have to prepare an appropriate data.frame. Look for existing \*TvAdmin.dbf\* files for necessary columns.

### Value

Five files will be created in "tv\_home/Data/databasename" directory. \*tvabund.dbf\* with occurrence information n long format, \*tvhabita.dbf\* with plot information, remarks.dbf with comments longer then 255 characters, \*TvAdmin.dbf\* with plot UUID's and tvwin.dbf with information about taxonomic reference list, and dictionary used.

### Author(s)

Florian Jansen @email florian.jansen@uni-rostock.de

**See Also**

tv.veg

---

`vegdata`*vegdata*

---

**Description**

This package provides a set of functions to load data from vegetation databases (at present Turboveg and vegetweb.de). Taxa can be (semi-)automatically be checked and adapted depending the scientific question. For this a hierachical taxonomic reference list is needed. Use [tv.veg()] to prepare data directly for further analyses. Set option `taxval` to TRUE, if your database is referenced with GermanSL or equivalent taxonomic reference list and you want to realize taxonomic checks and adaptations. For more details see `vignette('vegdata')`.

**References**

Jansen, F., Dengler, J (2011) Plant names in vegetation databases - a neglected source of bias, Journal of vegetation science, 21(6), 1179-1186. <http://dx.doi.org/10.1111/j.1654-1103.2010.01209.x>  
Jansen, Florian and Dengler, Juergen (2008) GermanSL - eine universelle taxonomische Referenzliste fuer Vegetationsdatenbanken, Tuexenia, 28, 239-253.

# Index

- \* **Area**
  - tv.site, 26
- \* **Survey**
  - tv.site, 26
- \* **Turboveg**
  - tv.bib, 21
  - tv.coverperc, 22
  - tv.db, 22
  - tv.home, 23
  - tv.obs, 24
  - tv.readXML, 25
  - tv.site, 26
  - tv.write, 30
- \* **###**
  - tv.site, 26
- \* **datasets**
  - lc.0, 8
  - lc.1, 8
  - lc.all, 9
  - taxlevels, 16
- \* **data**
  - elbaue, 7
- \* **manip**
  - tv.veg, 28
- \* **misc**
  - tv.veg, 28
- \* **package**
  - vegdata-package, 2
- child, 3
- comb.species, 4
- cwm, 4
- db\_download, 6
- db\_download\_eurosl (db\_download), 6
- db\_download\_germansl (db\_download), 6
- db\_path, 7
- elbaue, 7
- grep, 15
- lc.0, 8
- lc.1, 8
- lc.all, 9
- multipatt, 14
- parent, 9
- parse.taxa, 10
- print.syntab (syntab), 13
- recoding.taxa, 10
- sql\_collect, 11
- src\_eurosl (src\_vegdata), 11
- src\_germansl (src\_vegdata), 11
- src\_vegdata, 11
- syn, 12
- syntab, 13
- tax, 14, 19
- tax.refl, 15
- taxlevels, 16
- taxname.abbr, 16
- taxname.authors
  - (taxname.removeAuthors), 17
- taxname.removeAuthors, 17
- taxname.simpl, 17
- taxname.simplify (taxname.simpl), 17
- taxval, 18, 28, 29
- TCS.replace, 20
- tv.bib, 21
- tv.coverperc, 22, 29
- tv.db, 22
- tv.dict (tv.db), 22
- tv.eco (tv.traits), 27
- tv.home, 23, 24
- tv.metadata, 23
- tv.obs, 19, 24, 29, 30
- tv.readXML, 25
- TV.replace, 26
- tv.site, 25, 26, 29



tv.traits, [27](#)

tv.veg, [20](#), [25](#), [28](#)

tv.write, [30](#)

vegdata, [31](#)

vegdata-package, [2](#)