Package 'Rveg'

April 4, 2026

Title Digitization of Phytosociological Relevés

Version 0.1.9

Description Simple and fast tool for transforming phytosociological vegetation data into digital form for the following analysis. Danihelka, Chrtek, and Kaplan (2012, ISSN:00327786). Hennekens, and Schaminée (2001) <doi:10.2307/3237010>. Tichý (2002) <doi:10.1111/j.1654-1103.2002.tb02069.x>. Wickham, François, Henry, Müller (2022) <https://CRAN.R-project.org/package=dplyr>.

URL https://plant-ecology-lab-czu.com/rveg/

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

Imports utils, xml2

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Přemysl Král [aut, cre] (ORCID: <https://orcid.org/0000-0002-2887-651X>), Jan Douda [aut] (ORCID: <https://orcid.org/0000-0002-1205-364X>)

Maintainer Přemysl Král <kralp@fzp.czu.cz>

Repository CRAN

Date/Publication 2026-04-04 12:20:02 UTC

Contents

`addReleve` *addReleve: Digitize and Edit Relevés in an Rveg Database*

Description

The core interactive function of the `Rveg` package. `addReleve()` launches a console-based menu system that allows users to create new vegetation databases, digitize new relevés, and edit existing ones. It seamlessly manages both the species composition data (REL) and the environmental plot header data (HEAD). Parameter `start` was removed.

For list of commands in addReleve menu prompt `help` *or* `?`

Usage

```
addReleve(
  database = "NEW",
  save = "default",
  checklist = "default",
  customhead = NULL,
  extrahead = NULL,
  metadata = NULL
)
```

Arguments

`database`	Character. The path and name of an existing Rveg database (e.g., `"path/to/my_db"`). Defaults to `"NEW"` which creates a fresh database.
`save`	Character. The output path and name where the resulting database files (`*HEAD.csv` and `*REL.csv`) will be exported. Defaults to a temporary directory.
`checklist`	Character. The species checklist to use. Can be a built-in checklist (atm: `cz_dh2012`, `Czechia_slovakia_2015`, `cz_kaplan2019`, `wcvp_que`, `wcvp_por`) or a file path to a custom `txt` checklist. Default use `cz_dh2012`. see function `CreateChecklist`.
`customhead`	Character vector. A vector of strings defining completely custom header fields. Overrides the default schema.
`extrahead`	Character vector. Additional header fields to append to the end of the default or existing header schema.
`metadata`	Character vector of length 2. Used to store the Project Title and Project Description (e.g., `c("Alpine Flora", "Summer 2024 survey")`).

Value

Writes two linked CSV files (Rveg database) to the location specified by `save`: one containing the relevé species data (`*REL.csv`) and one containing the header data (`*HEAD.csv`).

Examples

```
if (interactive()) {
  # Launch the interactive menu for a new database
  addReleve(
    database = "NEW",
    save = "my_new_project",
    metadata = c("Project Title", "Project Description")
  )
}
```

`CreateChecklist` *CreateChecklist: Create a custom checklist for Rveg*

Description

Generates a custom species checklist formatted for use within the `Rveg` package. This function takes a list of full botanical names and deterministically converts them into unique, 7-character `ShortName` codes.

`Rveg` includes the following built-in checklists:

- `cz_dh2012` (Default: Checklist of vascular plants of the Czech Republic)
- `Czechia_slovakia_2015` (Turboveg compatible checklist)
- `cz_kaplan2019` (Rveg generated - Kaplan et al. 2019)
- `wcvp_que` (example WCVP subset for Quebec)
- `wcvp_por` (example WCVP subset for Portugal)

Usage

```
CreateChecklist(specieslist, export = "export")
```

Arguments

`specieslist`	Character vector OR Character. Either a vector of full botanical names in your R environment, or a path to a `.txt` file containing a single column named "FullName".
`export`	Character. The output path and name for the exported checklist. Defaults to a temporary directory.

Details

ShortName Generation Rules: The function guarantees a unique 7-character code for every species. The first 4 characters always represent a unique Genus code. The remaining 3 characters depend on the taxonomic rank:

- **Standard Species:** Genus (4) + first 3 letters of the specific epithet (e.g., `GENUEPI`).

- **Species (sp.):** Genus (4) + `-SP` (e.g., `GENU-SP`).
- **Hybrids (x):** Genus (4) + `*` + first 2 letters of hybrid epithet (e.g., `GENU*HY`).
- **Aggregates (agg.):** Genus (4) + `#` + first 2 letters of main epithet (e.g., `GENU#EP`).
- **Subspecies (ssp.):** Genus (4) + first letter of main epithet + `-` + first letter of infra epithet (e.g., `GENUE-I`).
- **Varieties/Forms (var./f.):** Genus (4) + first letter of main epithet + `;` + first letter of infra epithet (e.g., `GENUE;I`).
- **Sections (sect.):** Genus (4) + `SE` + first letter of section epithet (e.g., `GENUSEE`).

Note: If a 7-character code clash occurs, the 7th character is automatically adjusted using trailing letters or alphanumeric fallbacks to ensure absolute uniqueness.

Value

Writes a `.txt` file containing two columns (`ShortName` and `FullName`) to the location specified by `export`. The output can be then used inside `addReleve` and other functions.

Examples

```
# Example 1: Creating a checklist from a local txt file
CreateChecklist(
  specieslist = paste0(path.package("Rveg"), "/extdata/SpeciesList"),
)
```

`RvegCheck` *RvegCheck: Inspect, Validate, Edit and Repair an Rveg Database*

Description

A diagnostic and managing utility to verify the health and integrity of an `Rveg` database. `RvegCheck()` reads your database, checks for missing or corrupted data, and outputs a summary of the database's current state.

Usage

```
RvegCheck(database, export = "export", checklist = "default")
```

Arguments

`database`	Character. The path and name of the existing `Rveg` database to inspect (e.g., `"path/to/my_db"`).
`export`	Character. The output path and name where the modified database will be saved. Defaults to a temporary directory.
`checklist`	Character. The species checklist to validate the database against. By default, it uses the checklist defined in the database's metadata.

Details

Note: This function is currently under active development. #' At present, the function serves primarily as a diagnostic and metadata-repair tool. When run, it checks the database for a `project_name` and `project_description`. If these are missing, it will interactively prompt the user to supply them. It also prints vital database statistics, including the internal ID, creation date, last modification date, and any custom species added to the checklist.

In future updates, this function will also allow users to modify core database parameters, such as swapping the underlying species checklist or altering the header schema.

Value

Currently prints diagnostic information directly to the R console. If modifications are made, it exports the updated database files (`*HEAD.csv` and `*REL.csv`) to the path specified by `export`.

See Also

`RvegMerge` for combining databases, `RvegCombine` for manipulating data within a database.

Examples

```
if (interactive()) {
  # Inspect the built-in example database
  RvegCheck(
    database = file.path(path.package("Rveg"), "extdata/ExampleDB", "example_1")
  )
}
```

`RvegCombine` *RvegCombine: Merge Species or Vegetation Layers in an Rveg Database*

Description

An interactive utility that allows users to merge the abundance covers of specific species or entire vegetation layers within an existing `Rveg` database.

Usage

```
RvegCombine(database, export = "export", checklist = "default")
```

Arguments

`database`	Character. The path and name of the existing `Rveg` database to be modified (e.g., `"path/to/my_db"`).
`export`	Character. The output path and name where the modified database files (`*HEAD.csv` and `*REL.csv`) will be saved. Defaults to a temporary directory.
`checklist`	Character. The species checklist to be used. By default, it uses the checklist defined in the database's metadata.

Details

RvegCombine() operates via a console menu with two primary modes:

- **Layer Merging (**LAYER**):** Moves all species recorded in one specific layer (e.g., shrub layer 2) into another layer (e.g., tree layer 3).
- **Species Merging (**SPEC**):** Merges the records of one specific taxon into another across the entire database, which is highly useful for resolving taxonomic aggregates or correcting identification errors after data entry.

Mathematical Consolidation: #' When merging entities that both have non-zero percentage covers in the same relevé, the function does not simply add them together (which could exceed 100%). Instead, it uses a probabilistic sum formula to estimate the combined cover:

$$Combined = C_1 + C_2 \times (1 - \frac{C_1}{100})$$

Value

Writes two linked CSV files to the location specified by export, representing the modified Rveg database.

See Also

addReleve for data entry, RvegMerge for merging entire databases.

Examples

```
if (interactive()) {
  RvegCombine(
    database = file.path(path.package("Rveg"), "extdata/ExampleDB", "example_1")
  )
}
```

RvegLoad *RvegLoad: Load an Rveg Database into the R Environment*

Description

Reads an existing Rveg database (containing relevé data, header data, and metadata) into R for analysis. By default, it automatically translates the internal 7-character ShortName codes back into full botanical names and extracts the layer information.

Usage

```
RvegLoad(
  database = "default",
  checklist = "default",
  customscale = FALSE,
  variation = 1
)
```

Arguments

database	Character. The path and name of the Rveg database to load (e.g., "path/to/my_db"). Defaults to "default", which loads the package's built-in example database.
checklist	Character. The species checklist to use for taxonomy translation. By default, the function uses the checklist specified in the database's metadata.
customscale	Logical. If TRUE, launches an interactive prompt to convert custom abundance scales into numeric percentages. Defaults to FALSE.
variation	Numeric. Determines the format of the returned object (1, 2, or 3). Defaults to 1.

Details

The structure of the imported data is controlled by the variation parameter:

- variation = 1 **(Default):** Returns a single, large data frame. The header data (HEAD) is bound to the top of the species composition data (REL), with new columns added for FullName and layer.
- variation = 2**:** Returns the raw database as a list of three elements: $HeaderDATA, $RelDATA, and $meta (metadata) without any taxonomic translation.
- variation = 3**:** Returns the database as a list of three elements, but $RelDATA is processed to include the translated FullName and layer columns.

Custom Scales: If customscale = TRUE, the function operates interactively. It will pause and prompt the user to manually define percentage replacements for any custom abundance symbols found in the data.

Value

A data frame (if variation = 1) or a list containing data frames and metadata (if variation = 2 or 3).

Examples

```
# Example 1: Load the built-in Rveg database into a single data frame
my_data <- RvegLoad()

# Example 2: Load the database as a list with separated Header and Releve tables
my_list <- RvegLoad(variation = 3)
```

RvegMerge *RvegMerge: Merge Two Rveg Databases*

Description

Combines two separate Rveg databases into a single, unified database. This function cleanly merges both the species composition data (REL) and the environmental plot header data (HEAD), and seamlessly combines their metadata.

Usage

```
RvegMerge(database_1, database_2, export = "export")
```

Arguments

database_1	Character. The path and name of the first Rveg database (e.g., "path/to/db1").
database_2	Character. The path and name of the second Rveg database (e.g., "path/to/db2").
export	Character. The output path and name where the merged database files (*HEAD.csv and *REL.csv) will be saved. Defaults to a temporary directory.

Details

During the merge process:

- **Relevé Re-indexing:** The relevé columns from both databases are sequentially re-indexed (e.g., X1, X2, ..., Xn) to prevent column name collisions.
- **Checklist Validation:** The function checks the metadata of both databases to ensure they were built using the same species checklist. If the checklists differ, it will issue a warning, though the merge will still proceed.
- **Missing Species:** If a species exists in one database but not the other, the function automatically fills the absences with 0.
- **Metadata:** The total number of relevés is updated, and any custom species (extra_spec) from both databases are concatenated together.

Value

Writes two linked CSV files to the location specified by export, representing the combined Rveg database.

See Also

RvegCombine for manipulating data within a single database, addReleve for adding individual relevés.

Examples

```
# Example: Merging the built-in database with itself
db_path <- file.path(path.package("Rveg"), "extdata/ExampleDB", "example_1")

RvegMerge(
  database_1 = db_path,
  database_2 = db_path
)
```

RvegToJuice *RvegToJuice: Export an Rveg Database to JUICE compatible format*

Description

Exports an existing Rveg database into a format directly compatible with JUICE, a comprehensive software for vegetation classification. This function processes both the species composition data and the header data, formatting them to meet JUICE's import requirements.

Usage

```
RvegToJuice(database, export = "export", checklist = "default")
```

Arguments

database	Character. The path and name of an existing Rveg database to be exported (e.g., "path/to/my_db").
export	name of your exported csv file
checklist	Character. The species checklist to use. By default, the function attempts to read the checklist specified in the database's metadata. You can override this by providing a custom file path or a built-in dictionary string.

Details

To ensure seamless compatibility with JUICE, this function performs several background transformations:

- **Layer Mapping:** Rveg layers are automatically converted to JUICE's numeric layer representations (e.g., Tree layer "3" becomes "2", Shrub "2" becomes "4", Herb "1" becomes "6", etc.).
- **Absence Encoding:** Zero values (0) are converted to ..
- **Encoding:** Files are written using ISO-8859-15 encoding, which is the standard expected by JUICE for proper character rendering.

For JUICE import first import relevé data as Spreadsheet file and follow with Header data as Tab delimineted file.

Value

Writes two text-based CSV files to the location specified by `export`: one containing the header data (`*H.csv`) and one containing the relevé species data formatted with JUICE headers (`*R.csv`).

Examples

```
# Example: Exporting the built-in example Rveg database to JUICE format
RvegToJuice(
  database = file.path(path.package("Rveg"), "extdata/ExampleDB", "example_1")
)
```

`RvegToTv` *RvegToTv: Export an Rveg Database to Turboveg Compatible Format*

Description

Exports an existing `Rveg` database into a CSV format compatible with the Turboveg vegetation database management system. The function automatically reconstructs full botanical names from the internal `ShortName` codes and maps vegetation layers to standard Turboveg abbreviations.

Usage

```
RvegToTv(database, export = "export", checklist = "default", ver = 3)
```

Arguments

`database`	Character. The path and name of the existing `Rveg` database to be exported (e.g., `"path/to/my_db"`).
`export`	Character. The output path and name where the resulting Turboveg CSV file(s) will be saved. Defaults to a temporary directory.
`checklist`	Character. The species checklist used to match Rveg's 7-character `ShortName` codes back to their full botanical names. Defaults to `"default"`.
`ver`	Numeric. The target Turboveg version format to export to (either `2` or `3`). Defaults to `3`.

Details

During export, Rveg's alphanumeric layers are translated into Turboveg's specific layer codes (e.g., `0` becomes `ml`, `1` becomes `hl`, `2` becomes `s1`, `3` becomes `t1`, and `J` becomes `jl`).

The output structure changes depending on the target Turboveg version specified by the `ver` parameter:

- **Turboveg v3** (`ver = 3`)**:** Writes a single, combined `.csv` file containing both header and species data.
- **Turboveg v2** (`ver = 2`)**:** Writes two separate files: `*R.csv` for relevé species data and `*H.csv` for header data.

Value

Writes one or two CSV files to the location specified by `export`, depending on the chosen Turboveg version.

Examples

```
# Example: Exporting the built-in Rveg database to Turboveg v3 format
RvegToTv(
  database = file.path(path.package("Rveg"), "extdata/ExampleDB", "example_1"),
  ver = 3
)
```

`TvToRveg` *TvToRveg: Import Turboveg Data into an Rveg Database*

Description

Converts a Turboveg export file into a fully functional `Rveg` database. The function parses plot headers, standardizes species nomenclature against a specified checklist, and maps vegetation layers.

Usage

```
TvToRveg(tv, export = "export", checklist = "default", Rveglayers = TRUE)
```

Arguments

`tv`	Character. The file path to the Turboveg export file (`.csv` or `.xml`).
`export`	Character. The output path and name of the new `Rveg` database Defaults to a temporary directory.
`checklist`	Character. The species checklist used to match Turboveg full names to Rveg's 7-character `ShortName` codes. Defaults to `"default"`. Select one you want to use in your `Rveg` database or the one most similar to the one used in `Turboveg`.
`Rveglayers`	Logical. If `TRUE` (the default), Turboveg layer codes (e.g., 't', 's', 'h', 'm', 'j') are automatically translated into standard Rveg numeric/character layers (`3`, `2`, `1`, `0`, `J`).

Details

This function natively supports both `.csv` and `.xml` Turboveg export formats. In Turboveg, either selecet `Standard XML file` or `Spreadsheet table`. In the case of spreadsheet table, select format 'semicolon delimited and requested header data.

During the import process, the function operates interactively:

- **Species Resolution:** If a species in the Turboveg file cannot be automatically matched to the provided checklist, the function will pause and prompt the user to manually resolve the unknown species using a search interface.
- **Scale Selection:** The user will be prompted to specify whether the imported abundance data uses percentages ("P") or the Braun-Blanquet scale ("BB").

Value

Writes two linked CSV files (`*REL.csv` and `*HEAD.csv` = `Rveg database`) to the location specified by `export`.

Examples

```
if (interactive()) {
  # Example: Importing a Turboveg CSV export (or use tvexport.xml)
  TvToRveg(
    tv = file.path(path.package("Rveg"), "extdata/ExampleDB", "tvexport.csv"),
    Rveglayers = TRUE
  )
}
```

Index