TR8: Extract traits data for plant species

Gionata Bocci
Pisa (ITALY)
boccigionata@gmail.com

October 31, 2014

1 Rationale

The TR8 package has been built in order to provide the user with the possibility of easily retrieving traits data for plant species from the following publicly available databases:

Biolflor http://www2.ufz.de/biolflor/index.jsp [5]

Ecological Flora of the British Isles http://www.ecoflora.co.uk/ [3]

LEDA traitbase http://www.leda-traitbase.org/LEDAportal/ [4]

Ellenberg values for Italian Flora [6]

Flowering period for Italian Flora [6] (data retrieved from http://luirig.altervista.org/)

Mycorrhizal intensity database [2]

Please note that not all the traits available on the listed databases are downloaded by the package: this may change in future versions of the package (ie. some functionalities may be added and more traits will be made available).

2 Installation

The TR8 package is available on CRAN, thus it can be easily installed through:

```
> install.packages("TR8",dependencies = TRUE)
```

The option `dependencies = TRUE` takes care of installing the following packages (if they are not already installed) which are needed by TR8 to work properly:

- plyr[11]

- reshape[10]

- RCurl[7]

- XML[8]

- taxize[1]

- gWidgets[9]

- gWidgetstcltk

- rappdirs

Once the package is installed, you can load it with:

```
> library(TR8)
```

Please note that:

The user is asked to always cite the data sources: the development of traits databases is a long and costly process, thus all the users of the TR8 package are asked (and reminded **every time** they load the package) to always cite the original sources of the data (see paragraph 5).

2.1 Using the development version

The devel version of the package is hosted on github at `https://github.com/GioBo/TR8`: to use this version (instead of the stable one, released from CRAN), you'll need the `devtools` package (`https://github.com/hadley/devtools`):

```
> ## install the package
> install.packages("devtools")
> ## load it
> library(devtools)
> ## activate dev_mode
> dev_mode(on=T)
> ## install TR8
> install_github("GioBo/TR8")
> ## you can now work with TR8 functions
>
> ## if you want to go back and use the CRAN version
> ## already installed, simply deactivate dev_mode
> dev_mode(on=F)
```

3 Simple usage

Using the `TR8` package is fairly simple: users just need to call the `tr8` function passing, as arguments, a vector of plant species names (**withouth authors' names!**[1]) and a vector containing the codes corresponding to the traits which are to be downloaded:

```
> ## a vector containing a list of plant species names
> my_species<-c("Apium graveolens","Holcus mollis","Lathyrus sylvestris")
> ## a vector of traits
> to_be_downloaded<-c("reprod_B","strategy")
> ## now run tr8 and store the results in the my_traits object
> my_traits<-tr8(species_list = my_species,download_list = to_be_downloaded)
```

The codes which are accepted by TR8 are listed in the `available_tr8` database:

```
> ## see the firs lines of available_tr8 database
> head(available_tr8)
  short_code              description       db
1     h_max          Maximum height Ecoflora
2     h_min          Minimum height Ecoflora
3   le_area               Leaf area Ecoflora
4   le_long          Leaf longevity Ecoflora
5 phot_path Photosynthetic pathway Ecoflora
6   li_form               Life form Ecoflora
```

The database is composed of three columns:

short_code contains the codes that should be passed to the `download_list` argument of the `tr8` function.

description contains short description of each trait (please refer to the original sources for detailed descriptions).

db refers to the databases from which are providing traits data

Suppose the user is interested in downloading the *maximum height*, the *leaf area* and the *life form* (which are available through the *Ecolfora* database) for *Salix alba* and *Populus nigra* and store the resulting data in the `my_Data` object; the command should be:

```
> my_species<-c("Salix alba","Populus nigra")
> my_traits<-c("h_max","le_area","li_form")
> my_Data<-tr8(species_list = my_species, download_list = my_traits)
```

---

[1] This is needed since some traitbases do not include authors' names in the species' names.

The `tr8` function will take care of downloading the data and store them in the `my_Data` object; you can see the results by printing them:

```
> ## see the downloaded data
> print(my_Data)
```

Or you can convert them to a data frame using the `extract_traits` function:

```
> traits_dataframe<-extract_traits(my_Data)
```

All the traits are now contained in a data frame with species as rows and columns as traits; where no trait data were available, you will see a `NA`.

In order to make the dataframe more readable, traits' names (i.e. columns' names) are converted to shorter codes: to see a brief explanation of the codes used to identify the traits, use the `lookup` function:

```
> lookup(my_Data)
```

The object returned by the `lookup` function can also be stored in order to be available for further elaborations:

```
> my_lookup<-lookup(my_Data)
> head(my_lookup)
```

3.1 Interactive use of `tr8`

Up to now we've been using the `tr8` function in a non-interactive way. In order to help those user which are more familiar with a GUI approach, the function can also be run setting the `gui_config` parameter to `TRUE` (without providing any trait to the `download_list` parameter) and a multi-panel window will appear: the user is asked to choose those traits which are to be downloaded from the various databases [2].

For a detailed explanation of each level of a trait, please refer to the original websites (all the databases listed in the references provide the users with very precise and detailed descriptions).

Typically users will have a their vegetation data in the form of a *sites*species* dataframe (or matrix), thus they may want to extract traits data for the whole dataset (this time using the GUI to select traits), i.e. :

```
> ## suppose veg_data is our dataframe with
> ## plant species as columns and sites as rows
>
> ## extract species names
> specie_names<-names(veg_data)
```

---

[2]A note for Mac users: the GUI relies on the `Tcl/Tk` toolkit, thus if you want to run the GUI, please make sure that the `X11` package is installed - see "Tcl/Tk" issues at `http://cran.r-project.org/bin/macosx/RMacOSX-FAQ.html`

```
> ## use the tr8() function
> ## and tick those traits of interest in the pop-up window
> my_traits<-tr8(species_names,gui_config=TRUE)
> ## print the results
> print(my_traits)
```

3.2   Some notes on using `tr8`

A NOTE OF CAUTION: searching the web is a time (and Internet band) consuming activity, thus the higher the number of your plant species and the traits to be retrieved, the longer it will take to `tr8()` to complete its job. Moreover, in order not to overflow the remote databases with `http` requests, the `tr8` function will always pause between one search and the following one.

   A (SECOND) NOTE OF CAUTION: some users adopt the following workflow for analysing their vegetation data:

1. insert vegetation data into a *spreadsheet file* with species as columns' and sites' as rows

2. export the spreadsheet file as a `.csv` file

3. import the `.csv` file into a **R** dataframe.

   When following these steps, a dot (".") will be inserted between Genus and Species of each plant species name (i.e. column names in the `R` dataframe will not be in the form `c("Abies alba", "Salix alba")` but in the form `c("Abies.alba", "Salix.alba")`). This may cause problems for further processing of plants' species names, thus, in order to avoid this problem, please use the `check.names=F` option in `read.csv`. E.g. suppose that `my_veg_data.csv` is the `csv` file: in the **R** console, one should use:

```
> My_data<-read.csv("my_veg_data.csv",
+                    header=T,row.names=1,check.names=F)
```

4   Interpreting retrieved data

Please note that for many traits there is more than one entry in the original databases: in those cases, in order to obtain a single value the following strategy was adopted:

Quantitative traits the mean of all the values was calculated (e.g. when multiple values for "Seed weight mean" are available, the mean of these value is calculated)

Qualitative traits all the values are taken into account and "joined" together in a single string (the values are separated by a score "−")

Nota bene: in some cases some traits are stored as *string* in the original databases, even though they should be treated as numbers (e.g. the number *five* is stored as a string - i.e. "5", not as the numeric value 5): in those case `tr8` function is not able to interpret that entry as a numeric, thus, applying the above mentioned criteria to merge multiple traits, strange outputs may result from `tr8`, e.g. if a species has two entries for the trait `height` - day 3 and 3.5 meters - the merged value will not be the numeric mean (3.25) but the union of the two strings ("3-3.5").

5   Citing sources of information

Users of the `TR8` package should always cite the sources of information which provided the traits data: the correct citations to be used for the retrieved data can be obtained through the `bib` method; just use:

```
> bib(my_traits)
```

6   Suggested workflow

We strongly suggest to always check plant species names with the `tnrs` function (from the `taxize` package) before using the `tr8` function; thus a typical workflow would be the following:

1. Check plant species names (e.g. with something like the following - please refer to the `taxize` package documentation[1] for further details)

   ```
   > species_names<-names(veg_data)
   > checked_names<-tnrs(species_names,source="iPlant_TNRS")
   > print(checked_names)
   ```

   Check which species (rows) in the table have a "score" value lower than 1 and check their names; if needed, correct them before using the tr8() function.

2. Run `tr8` (in this case using the GUI):

   ```
   > my_traits<-tr8(species_names,gui_config = TRUE)
   > print(my_traits)
   ```

3. You may want to have these traits available as a data frame: just use the `extract_traits` function which uses the results of `tr8` (in this case it's the `my_traits` objects) and returns a data frame.

```
> traits_df<-extract_traits(my_traits)
```

4. Observing a big data frame inside R could be difficult, thus users may want to save the `traits_df` data frame as a `.csv` file:

```
> save(traits_df,file="traits_df.csv")
```

and then open that file with a spreadsheet software (e.g. LibreOffice).

7 Further steps

The `TR8` package comes with another vignette, called `TR8_workflow`, which shows a typical workflow describing all the steps needed for retrieving and analysing traits data with `tr8`, listing the most common problems that could be faced and the possible solutions to fix them.
The vignette can be opened from within R, using:

```
> vignette("TR8_workflow")
```

Another vignette (called `Expanding_TR8`) shows to programmers how sources of data can be added to `TR8` (i.e. how functions for retrieving data should be written so that they can be easily integrated in `TR8`).

8 Local storage of LEDA and Akhmetzhanova data

The LEDA Traitbase datafiles and Akhmetzhanova database are text files (either `.txt` or `.csv` files) which are available for download at the LEDA (`http://www.leda-traitbase.org/LEDAportal/data_files.jsp`) and at the *Ecological Archives* websites . These files are (quite) big in size, thus downloading them every time the `tr8()` function is used is a time consuming activity[3]. In order to make data retrieval more efficient, when `tr8` is run for the first time requiring traits data contained in the LEDA or in the Akhmetzhanova databases, these data are downloaded and a copy is stored in a local directory and made available to future requests[4].

---

[3]The text files are not distributed together with the `TR8` package - which would save time and memory when executing the `tr8()` function - in order to avoid possible licensing conflicts between the `TR8`' GPL license and these datasets.

[4]By default these files will be installed in the directories which are commonly used for storing applications' data (which depends on the underlying operating systems; see `https://github.com/hadley/rappdirs` for details).

References

[1] Chamberlain, S. and Szocs, E. taxize - taxonomic search and retrieval in r. *F1000Research*, 2013.

[2] A.A. Akhmetzhanova, N.~A. Soudzilovskaia, V.~G. Onipchenko, W.K. Cornwell, V.A. Agafonov, I.A. Selivanov, and J.~H.C. Cornelissen. A rediscovered treasure: mycorrhizal intensity database for 3000 vascular plant species across the former Soviet Union: Ecological archives e093-059. *Ecology*, 93(3):689–690, 2012.

[3] A.~H. Fitter and H.~J. Peat. The Ecological Flora Database. *J. Ecol*, 82:415–425, 1994.

[4] M.~Kleyer, R.~M. Bekker, I.~C. Knevel, J.~P. Bakker, K.~Thompson, M.~Sonnenschein, P.~Poschlod, J.~M. van Groenendael, L.~Klimes, J.~Klimesova, S.~Klotz, G.~M. Rusch, M.~Hermy, D.~Adriaens, G.~Boedeltje, B.~Bossuyt, A.~Dannemann, P.~Endels, L.~Götzenberger, J.~G. Hodgson, A-K. Jackel, I.~Kühn, D.~Kunzmann, W.~A. Ozinga, C.~Römermann, M.~Stadler, J.~Schlegelmilch, H.~J. Steendam, O.~Tackenberg, B.~Wilmann, J.~H.~C. Cornelissen, O.~Eriksson, E.~Garnier, and B.~Peco. The LEDA Traitbase: a database of life-history traits of the Northwest European flora. *JOURNAL OF ECOLOGY*, 96(6):1266–1274, 2008.

[5] S.~Klotz, I.~Kühn, and W.~Durka. BIOLFLOR - Eine Datenbank zu biologisch-ökologischen Merkmalen zur Flora von Deutschland. Schriftenreihe für Vegetationskunde 38: 1-333. (Bundesamt für. Bonn, Bundesamt für Naturschutz). *Schriftenreihe für Vegetationskunde*, 38:1–333, 2002.

[6] S.~Pignatti, P.~Menegoni, and S.~Pietrosanti. Biondicazione attraverso le piante vascolari. Valori di indicazione secondo Ellenberg (Zeigerwerte) per le specie della Flora d'Italia. *Braun-Blanquetia*, 39:97, 2005.

[7] D.~Temple~Lang. *RCurl: General network (HTTP/FTP/...) client interface for R*, 2013. R package version 1.95-4.1.

[8] D.~Temple~Lang. *XML: Tools for parsing and generating XML within R and S-Plus.*, 2013. R package version 3.98-1.1.

[9] J.. Based on the iwidgets code of Simon~Urbanek Verzani, suggestions~by Simon~Urbanek, Philippe Grosjean, and Michael Lawrence. *gWidgets: gWidgets API for building toolkit-independent, interactive GUIs*, 2012. R package version 0.0-52.

[10] H.~Wickham. Reshaping data with the reshape package. *Journal of Statistical Software*, 21(12), 2007.

[11] H.~Wickham. The split-apply-combine strategy for data analysis. *Journal of Statistical Software*, 40(1):1–29, 2011.