TR8: Extract traits data for plant species

Gionata Bocci
Pisa (ITALY)
boccigionata@gmail.com

October 12, 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]

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]

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).

3 Simple usage

Using the TR8 package is fairly simple: users just need to call the tr8 function passing, as argument, a vector of plant species names:

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

A multi-panel window will appear: the user is asked to choose those traits which are to be downloaded from the various databases; if data from the LEDA Traitbase are selected, after clicking OK a second GUI will appear, asking if a local copy of the LEDA .txt files has already been downloaded (see paragraph 7 for a more detailed explanation of the way TR8 deals with the LEDA Traitbase): if this is the case, please tick the ”yes” button, click ”Ok” and then choose the folder which contains the already downloaded files. The tr8 function will take care of downloading the data and store them in the my_traits object; you can see the results by printing them:

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

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

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

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 (ie. 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_traits)
```

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).

Tipically 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, ie.:

```
> ## suppose veg_data is our dataframe with
> ## plant species as columns and sites as rows
>
> ## extract species names
> specie_names<-names(veg_data)
> ## use the tr8() function
> my_traits<-tr8(species_names)
> ## print the results
> print(my_traits)
```

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, we suggest to split the the list of plant species in smaller chunks.

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`. Eg. 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 (eg. 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 ”−”)

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 usage

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 (eg. with something like the following - please refere 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`

   ```
   > my_traits<-tr8(species_names)
   > 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 and open that with a spreadsheet software (eg. LibreOffice)

7 LEDA Traitbase

The LEDA Traitbase datafiles are `.txt` files which are available for download at the LEDA website (`http://www.leda-traitbase.org/LEDAportal/data_files.jsp`). These files are (quite) big in size, thus downloading them every time the `tr8()` function is used is a time consuming activity[1]. We thus suggest the users to run the `leda_download_to_local_directory()`[2] function once to store a local copy of the LEDA datafiles and use that local copy every time the `tr8()` function is run.

```
> ## run the function
> leda_download_to_local_directory()
> ## and select a folder where the datafiles will be
> ## downloaded (remeber this folder, since you will be later
> ## ask for that when running the tr8() function)
```

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.

[1] The .txt 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`' license and LEDA datasets.

[2] The name is quite self−explanatory...

[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.