WEB TOOL

UPDATED taxize: taxonomic search and retrieval in R [v2; ref status: indexed, http://f1000r.es/24v]

Scott A. Chamberlain[1*], Eduard Szöcs[2*]

[1]Biology, Simon Fraser University, Burnaby, Canada
[2]Institute for Environmental Sciences, University Koblenz-Landau, Landau, Germany

[*] Equal contributors

v2 **First Published:** 18 Sep 2013, **2**:191 (doi: 10.12688/f1000research.2-191.v1)
Latest Published: 28 Oct 2013, **2**:191 (doi: 10.12688/f1000research.2-191.v2)

Abstract

All species are hierarchically related to one another, and we use taxonomic names to label the nodes in this hierarchy. Taxonomic data is becoming increasingly available on the web, but scientists need a way to access it in a programmatic fashion that's easy and reproducible. We have developed taxize, an open-source software package (freely available from http://cran.r-project.org/web/packages/taxize/index.html) for the R language. taxize provides simple, programmatic access to taxonomic data for 13 data sources around the web. We discuss the need for a taxonomic toolbelt in R, and outline a suite of use cases for which taxize is ideally suited (including a full workflow as an appendix). The taxize package facilitates open and reproducible science by allowing taxonomic data collection to be done in the open-source R platform.

Article Status Summary

Referee Responses

Referees	1	2	3
v1 published 18 Sep 2013	✓ report 1	✓ report 1	✓ report 1
v2 published 28 Oct 2013 UPDATED	✓ report 1	✓ report 1	✓ report 1

1 **Will Pearse**, University of Minnesota USA

2 **Gavin Simpson**, University of Regina Canada

3 **Ethan White**, Utah State University USA

Latest Comments

No Comments Yet

Corresponding author: Scott A. Chamberlain (myrmecocystus@gmail.com)

How to cite this article: Chamberlain SA, Szöcs E (2013) taxize: taxonomic search and retrieval in R [v2; ref status: indexed, http://f1000r.es/24v] *F1000Research* 2013, **2**:191 (doi: 10.12688/f1000research.2-191.v2)

Grant information: The author(s) declared that no grants were involved in supporting this work.

Competing Interests: No competing interests were disclosed.

First Published: 18 Sep 2013, **2**:191 (doi: 10.12688/f1000research.2-191.v1)
First Indexed: 23 Sep 2013, **2**:191 (doi: 10.12688/f1000research.2-191.v1)

UPDATED **Changes from Version 1**

We thank the reviewers for their comments. In addition to our responses to reviewers at the bottom, a detailed listing of reviewer comments and our discussion about them can be found here: https://github.com/ropensci/taxize_/issues/178. The following is a summary of the changes made in response to reviewer's comments:

* We have improved language where pointed out by reviews: removed sentences, changed awkward language, and corrected spelling and grammar mistakes.

* In response to comments by two of the three reviewer's we have added a third appendix that goes over using API keys and how to install the development version of the software.

* There were a number of suggestions about changing the software itself (improving naming of functions and outputs). We agree with these suggestions, and although the changes to the software have not been made yet, we plan on making the changes in the next version of the software and we will then update the manuscript accordingly.

See referee reports

Introduction

Evolution by natural selection has led to a hierarchical relationship among all living organisms. Thus, species are categorized using a taxonomic hierarchy, starting with the binomial species name (e.g, *Homo sapiens*), moving up to genus (*Homo*), then family (*Hominidae*), and on up to Domain (*Eukarya*). Although taxonomic classifications are human constructs created to understand the real phylogeny of life[1], they are nonetheless essential to organize the vast diversity of organisms. Biologists, whether studying organisms at the cell, organismal, or community level, can put their study objects into taxonomic context, allowing them to infer close and distant relatives, find relevant literature, and more.

The use of taxonomic names is, unfortunately, not straightforward. Taxonomic names often vary due to name revisions at the generic or specific levels, lumping or splitting lower taxa (genera, species) among higher taxa (families), and name spelling changes. For example, a study found that a compilation of 308,000 plant observations from 51 digitized herbarium records had 22,100 unique taxon names, of which only 13,000 were accepted names[2,3]. In addition, there is no one authoritative source of taxonomic names for all taxa - although, there are taxon specific sources that are used by many scientists. Different sources (e.g., uBio [Universal Biological Indexer and Organizer], Tropicos, ITIS [Integrated Taxonomic Information Service]) may use different accepted names for the same taxon. For example, while ITIS has *Helianthus x glaucus* as an accepted name, The Plant List (http://www.theplantlist.org) gives that name as unresolved. But *Helianthus glaucus* is an accepted name in The Plant List, while ITIS does not list this name.

One attempt to help inconsistencies in taxonomy is the use of numeric codes. For example, ITIS assigns a Taxonomic Serial Number (TSN) to each taxon, while uBio assigns each taxon a NameBank identifier (namebankID), and Tropicos assigns their own identifier to each taxon. Codes are helpful within a database as they can easily refer to, for example, *Helianthus annuus* with a code like 123456 instead of its whole name. However, each database uses their own code; in this case for *Helianthus annuus*, ITIS uses 36616, uBio uses 2658020, and Tropicos uses 40022652. As there are no universal codes for taxa across databases, this can lead to additional confusion. Last, name comparisons across databases have to be done with the actual names, not the codes.

Taxonomic data is getting easier to obtain through the web (e.g., http://eol.org/). However, there are a number of good reasons to obtain taxonomic information programatically rather than through a web interface. First, if you have more than a few names to look up on a website, it can take quite a long time to enter each name, get data, and repeat for each species. Programatically getting taxonomic names solves the problem by looping over a list of names. In addition, doing taxonomic searching, etc. becomes reproducible. With increasing reports of irreproducibility in science[4,5], it is extremely important to make science workflows repeatable.

The R language is widely used by biologists, and now has over 5,000 packages on the Comprehensive R Archive Network (CRAN) to extend R. R is great for manipulating, visualizing and fitting statistical models to data. Gentleman *et al.*[6] give a detailed discussion of advantages of R in computational biology. Getting data from the web will be increasingly common as more and more data gets moved to the cloud. Therefore, there is a need to get data from the web directly into R. Increasingly, data is available from the web via application programming interfaces (API). These allow computers to talk to one another using code that is not human readable, but is machine readable. Web APIs often define a number of methods that allow users to search for a species name, or retrieve the synonyms for a species name, for example. A further advantage of APIs is that they are language agnostic, meaning that data can be consumed in almost any computing context, allowing users to interact with the web API without having to know the details of the code. Moreover data can be accessed from every computer, whereas for example an Excel file can only be opened in a few programs.

The goal of taxize is to make many use cases that involve retrieving and resolving taxonomic names easy and reproducible. In taxize, we have written a suite of R functions that interact with many taxonomic data sources via their web APIs (Table 1). The interface to each function is usually a simple list of species names, just as a user would enter when interacting with a website. Therefore, we hope that moving from a web to an R interface for taxonomic names will be relatively seamless (if one is already nominally familiar with R).

Here, we justify the need for programmatic taxonomic resolution tools like taxize, discuss our data sources, and run through a suite of use cases to demonstrate the variety of ways that users can use taxize.

Why do we need taxize?

There is a large suite of applications developed around the problem of searching for, resolving, and getting higher taxonomy for species

Table 1. Some key functions in taxize, what they do, and their data sources.

Function name	What it does	Source
apg_lookup	Changes names to match the APGIII list	Angiosperm Phylogeny Group http://www.mobot.org/MOBOT/research/APweb/
classification	Upstream classification	Various
col_children	Direct children	Catalogue of Life http://www.catalogueoflife.org/
col_downstream	Downstream taxa to specified rank	Catalogue of Life http://www.catalogueoflife.org/
eol_hierarchy	Upstream classification	Encyclopedia of Life http://eol.org/
eol_search	Search EOL taxon information	Encyclopedia of Life http://eol.org/
get_seqs	Get NCBI sequences	National Center for Biotechnology Information[7]
get_tsn	Get ITIS TSN	Integrated Taxonomic Information System http://www.itis.gov/
get_uid	Get NCBI UID	National Center for Biotechnology Information[7]
gisd_isinvasive	Invasiveness status	Global Invasive Species Database http://www.issg.org/database/welcome/
gni_parse	Parse scientific names into components	Global Names Index http://gni.globalnames.org/
gni_search	Search EOL's global names index	Global Names Index http://gni.globalnames.org/
gnr_resolve	Resolve names using EOL's global names index	Global Names Resolver http://resolver.globalnames.org/
itis_downstream	Downstream taxa to specified rank	Integrated Taxonomic Information System http://www.itis.gov/
iucn_status	IUCN status	IUCN Red List http://www.iucnredlist.org
phylomatic_tree	Get a plant Phylogeny	Phylomatic[8]
plantminer	Search Plantminer	Plantminer[9]
searchbycommonname	Search ITIS by common name	Integrated Taxonomic Information System http://www.itis.gov/
searchbyscientificname	Search ITIS by scientific name	Integrated Taxonomic Information System http://www.itis.gov/
tax_name	Get taxonomic name for specific rank	Various
tax_rank	Get rank of a taxonomic name	Various
tnrs	Resolve names using iPlant	iPlant Taxonomic Name Resolution Service http://tnrs.iplantcollaborative.org/
tp_acceptednames	Check for accepted names using Tropicos	Tropicos http://www.tropicos.org/
tpl_search	Search the Plant List	The Plant List http://www.theplantlist.org
ubio_namebank	Search uBio	uBio http://www.ubio.org/index.php?pagename=sample_tools

names. For example, Linnaeus http://linnaeus.sourceforge.net/ provides the ability to search for taxonomic names in documents and normalize those names found. In addition, there are many web interfaces to search for and normalize names such as Encyclopedia of Life's Global Names Resolver http://resolver.globalnames.org/, uBio tools http://www.ubio.org/index.php?pagename=sample_tools, and iPlant's Taxonomic Name Resolution Service http://tnrs.iplantcollaborative.org/.

All of these data repositories provide ways to search for taxonomic names and resolve them in some cases. However, scientists ideally need a tool that is free and can be used programmatically, thereby facilitating reproducible research. The goal of taxize is to facilitate the creation of reproducible and easy to use workflows for searching for taxonomic names, resolving them, getting higher taxonomic names, and other tasks related to research dealing with species.

Data sources and package details

taxize uses many data sources (Table 1), and more can be easily added. There are two common tasks provided by the data sources: name search and name resolution. Other functionality in taxize includes retrieving a classification tree for a species, or retrieving child taxa of a focal taxon. One of the data sources (Phylomatic) returns phylogenies, while another (NCBI) returns genetic sequence data. However, there are other R packages that are focused solely on sequence data, such as rsnps[10], rentrez[11], BoSSA[12], and ape[13], so taxize does not venture deeply into these other domains.

Some of the data sources taxize interacts with require authentication. That is, in addition to the search terms the user provides (e.g., *Homo sapiens*), the data provider requires an alphanumeric identification key. This is necessary in some cases so that API providers can 1) better prevent databases crashing from too many requests, 2) collect analytics on requests to their API to provide better performance, etc., and 3) provide user level modification of rules for interacting with the API. The services that require an API key in taxize are: Encyclopedia of Life (EOL) http://eol.org/, the Universal Biological Indexer and Organizer (uBio) http://www.ubio.org/index.php?pagename=sample_tools, Tropicos http://www.tropicos.org/, and Plantminer[9]. One can easily obtain API keys by visiting the website of each service (see

Table 1 for links to each site). There are two typical ways of using API keys. First, you can pass in your API key in a function call (e.g., *ubio_namebank(srchName='Ursus americanus', key='your_alphanumeric_key')*). Second, you can store your key in the .Rprofile file, which is a common place to store settings. We recommend the second option as it simplifies function calls as taxize detects the stored keys.

taxize would not have been possible without the work of others. taxize uses httr[14] and RCurl[15] for performing calls to web APIs, XML[16] for parsing XML, RJSONIO[17] for parsing JSON, and stringr[18] and plyr[19] for manipulating data.

New data sources can be added; for example, we plan to add the following sources: Wikispecies and The Tree of Life. A connection to www.freshwaterecology.info (a database with autecological characteristics, ecological preferences and biological traits as well as distribution patterns of more than 12,000 European freshwater organisms belonging to fish, macro-invertebrates, macrophytes, diatoms and phytoplankton) will be finished when their new API is released. In addition, the authors welcome further suggestions of data sources to be added.

Use cases

First, install taxize

First, one must install and load taxize into the R session.

```
install.packages("taxize")
library(taxize)
```

Advanced users can also download and install the latest development copy from GitHub https://github.com/ropensci/taxize_, also permanently available at http://dx.doi.org/10.5281/zenodo.7097.

Resolve taxonomic names

This is a common task in biology. We often have a list of species names and we want to know a) if we have the most up to date names, b) if our names are spelled correctly, and c) the scientific name for a common name. One way to resolve names is via the Global Names Resolver (GNR) service provided by the Encyclopedia of Life http://resolver.globalnames.org/. Here, one can search for two misspelled names:

```
temp <- gnr_resolve(names = c("Helianthos annus",
                              "Homo saapiens"))
temp[ , -c(1,4)]

#                matched_name       data_source_title
# 1      Helianthus annuus L.       Catalogue of Life
# 2          Helianthus annus GBIF Taxonomic Backbone
# 3          Helianthus annus                     EOL
# 4       Helianthus annus L.                     EOL
# 5          Helianthus annus           uBio NameBank
# 6 Homo sapiens Linnaeus, 1758       Catalogue of Life
```

The correct spellings are *Helianthus annuus* and *Homo sapiens*. Another approach uses the Taxonomic Name Resolution Service via the Taxosaurus API http://taxosaurus.org/ developed by iPLant and the Phylotastic organization. In this example is a list of species names, some of which are misspelled, and then call the API with the *tnrs* function.

```
mynames <- c("Helianthus annuus", "Pinus contort",
             "Poa anua","Abis magnifica", "Rosa california",
             "Festuca arundinace","Sorbus occidentalos",
             "Madia sateva")
tnrs(query = mynames) [ , -c(5:7)]

#        submittedName         acceptedName     sourceId score
# 9  Helianthus annuus     Helianthus annus  iPlant_TNRS     1
# 10 Helianthus annuus     Helianthus annus         NCBI     1
# 4      Pinus contort       Pinus contorta  iPlant_TNRS  0.98
# 5           Poa anua            Poa annua  iPlant_TNRS  0.96
# 3      Abis magnifica      Abies magnifica  iPlant_TNRS  0.96
# 7    Rosa california     Rosa californica  iPlant_TNRS  0.99
# 8    Rosa california          California         NCBI     1
# 2 Festuca arundinace  Festuca arundinacea  iPlant_TNRS  0.99
# 1 Sorbus occidentalos  Sorbus occidentalis  iPlant_TNRS  0.99
# 6       Madia sateva         Madia sativa  iPlant_TNRS  0.97
```

It turns out there are a few corrections: e.g., *Madia sateva* should be *Madia sativa*, and *Rosa california* should be *Rosa californica*. Note that this search worked because fuzzy matching was employed to retrieve names that were close, but not exact matches. Fuzzy matching is only available for plants in the TNRS service, so we advise using EOL's Global Names Resolver if you need to resolve animal names.

taxize takes the approach that the user should be able to make decisions about what resource to trust, rather than making the decision on behalf of the user. Both the EOL GNR and the TNRS services provide data from a variety of data sources. The user may trust a specific data source, and thus may want to use the names from that data source. In the future, we may provide the ability for taxize to suggest the best match from a variety of sources.

Another common use case is when there are many synonyms for a species. In this example, there are six synonyms of the currently accepted name for a species.

```
library(plyr)
mynames <- c("Helianthus annuus ssp. jaegeri",
             "Helianthus annuus ssp. lenticularis",
             "Helianthus annuus ssp. texanus",
             "Helianthus annuus var. lenticularis",
             "Helianthus annuus var. macrocarpus",
             "Helianthus annuus var. texanus")
tsn <- get_tsn(mynames)
ldply(tsn, itis_acceptname)

#   submittedTsn          acceptedName   acceptedTsn
# 1       525928      Helianthus annuus         36616
# 2       525929      Helianthus annuus         36616
# 3       525930      Helianthus annuus         36616
# 4       536095      Helianthus annuus         36616
# 5       536096      Helianthus annuus         36616
# 6       536097      Helianthus annuus         36616
```

Retrieve higher taxonomic names

Another task biologists often face is getting higher taxonomic names for a taxa list. Having the higher taxonomy allows you to put into context the relationships of your species list. For example, you may find out that species A and species B are in Family C, which may lead to some interesting insight, as opposed to not knowing that Species A and B are closely related. This also makes it easy to aggregate/standardize data to a specific taxonomic level

(e.g., family level) or to match data to other databases with different taxonomic resolution (e.g., trait databases).

A number of data sources in taxize provide the capability to retrieve higher taxonomic names, but we will highlight two of the more useful ones: Integrated Taxonomic Information System (ITIS) http://www.itis.gov/ and National Center for Biotechnology Information (NCBI)[7]. First, search for two species, *Abies procera* and *Pinus contorta* within ITIS.

```
specieslist <- c("Abies procera", "Pinus contorta")
classification(specieslist, db = "itis")

# $`Abies procera`
#         rankName           taxonName          tsn
# 1        Kingdom             Plantae       202422
# 2     Subkingdom      Viridaeplantae       846492
# 3   Infrakingdom         Streptophyta       846494
# 4       Division         Tracheophyta       846496
# 5    Subdivision     Spermatophytina       846504
# 6  Infradivision        Gymnospermae       846506
# 7          Class           Pinopsida       500009
# 8          Order             Pinales       500028
# 9         Family            Pinaceae        18030
# 10         Genus               Abies        18031
# 11       Species       Abies procera       181835
#
# $`Pinus contorta`
#          rankName           taxonName          tsn
# 1          Kingdom             Plantae       202422
# 2       Subkingdom      Viridaeplantae       846492
# 3     Infrakingdom         Streptophyta       846494
# 4         Division         Tracheophyta       846496
# 5      Subdivision     Spermatophytina       846504
# 6    Infradivision        Gymnospermae       846506
# 7            Class           Pinopsida       500009
# 8            Order             Pinales       500028
# 9           Family            Pinaceae        18030
# 10           Genus               Pinus        18035
# 11         Species       Pinus contorta       183327
```

It turns out both species are in the family Pinaceae. You can also get this type of information from the NCBI by excuting the following code in R: *classification(specieslist, db = 'ncbi')*.

Instead of a full classification, you may only want a single name, say a family name for your species of interest. The function *tax_name* is built just for this purpose. As with the *classification*-function you can specify the data source with the *db* argument, either ITIS or NCBI.

```
tax_name(query = "Helianthus annuus", get = "family",
         db = "itis")

#       family
# 1 Asteraceae

tax_name(query = "Helianthus annuus", get = "family",
         db = "ncbi")

#       family
# 1 Asteraceae
```

If a data source does not provide information on the queried species, the result could be taken from another source and the results from the different sources could be pooled.

Interactive name selection

As mentioned previously most databases use a numeric code to reference a species. A general workflow in taxize is: Retrieve Code for the queried species and then use this code to query more data/information. Below are a few examples. When you run these examples in R, you are presented with a command prompt asking for the row that contains the name you would like back; that output is not printed below for brevity. In this example, the search term has many matches. The function returns a data.frame of the matches, and asks for the user to input which row number to accept.

```
get_tsn(searchterm = "Heliastes", searchtype = "sciname")

#                combinedname          tsn
# 1         Heliastes bicolor       615238
# 2       Heliastes chrysurus       615250
# 3         Heliastes cinctus       615573
# 4      Heliastes dimidiatus       615257
# 5      Heliastes hypsilepis       615273
# 6     Heliastes immaculatus       615639
# 7     Heliastes opercularis       615300
# 8           Heliastes ovalis       615301
#  1
# NA
# attr(,"class")
# [1] "tsn"
```

In another example, you can pass in a long character vector of taxonomic names:

```
splist <- c("annona cherimola", 'annona muricata',
            "quercus robur", "shorea robusta",
            "pandanus patina","oryza sativa",
            "durio zibethinus")
get_tsn(searchterm = splist, searchtype = "sciname")

# [1] "506198" "18098" "19405" "506787" "507376" "41976"
# [7] "506099"
# attr(,"class")
# [1] "tsn"
```

In another example, note that no match at all returns an NA:

```
get_uid(sciname = c("Chironomus riparius", "aaa vva"))

# [1] "315576" NA
# attr(,"class")
# [1] "uid"
```

Retrieve a phylogeny

Ecologists are increasingly taking a phylogenetic approach to ecology, applying phylogenies to topics such as the study of community structure[20], ecological networks[21], and functional trait ecology[22]. Yet, many biologists are not adequately trained in reconstructing phylogenies. Fortunately, there are some sources for getting a phylogeny without having to know how to build one; one of these is for angiosperms, called Phylomatic[8]. We have created a workflow in taxize that accepts a species list, and taxize works behind the scenes to get higher taxonomic names, which are required by Phylomatic to get a phylogeny. Here is a short example, producing the tree in Figure 1.

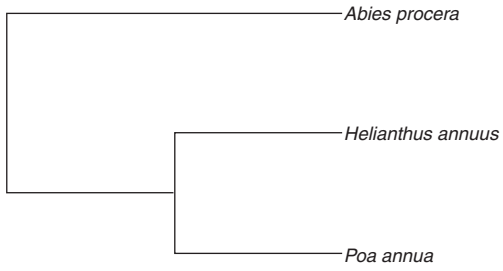

Figure 1. A phylogeny for three species. This phylogeny was produced using the *phylomatic_tree* function, which queries the Phylomatic database, and prunes a previously created phylogeny of plants.

```
taxa <- c("Poa annua", "Abies procera", "Helianthus annuus")
tree <- phylomatic_tree(taxa = taxa)
tree$tip.label <- capwords(tree$tip.label)
plot(tree, cex = 1)
```

Behind the scenes the function *phylomatic_tree* retrieves a Taxonomic Serial Number (TSN) from ITIS for each species name, then a string is created for each species like this *poaceae/oryza/oryza_sativa* (with format "family/genus/genus_epithet"). These strings are submitted to the Phylomatic API, and if no errors occur, a phylogeny in newick format is returned. The *phylomatic_tree()* function also cleans up the newick string and converts it to an ape *phylo* object, which can be used for plotting and phylogenetic analyses. Be aware that Phylomatic has certain limitations - refer to the paper describing Phylomatic[8] and the website http://phylodiversity.net/phylomatic/.

What taxa are children of the taxon of interest?

If someone is not a taxonomic specialist on a particular taxon they probably do not know what children taxa are within a family, or within a genus. This task becomes especially unwieldy when there are a large number of taxa downstream. You can of course go to a website like Wikispecies http://species.wikimedia.org/wiki/Main_Page or Encyclopedia of Life http://eol.org/ to get downstream names. However, taxize provides an easy way to programatically search for downstream taxa, both for the Catalogue of Life (CoL) http://www.catalogueoflife.org/ and the Integrated Taxonomic Information System http://www.itis.gov/. Here is a short example using the CoL in which we want to find all the species within the genus *Apis* (honey bees).

```
col_downstream(name = "Apis", downto = "Species")[[1]]

#  childtaxa_id        childtaxa_name  childtaxa_rank
# 1      6971712    Apis andreniformis         Species
# 2      6971713           Apis cerana         Species
# 3      6971714          Apis dorsata         Species
# 4      6971715            Apis florea         Species
# 5      6971716    Apis koschevnikovi         Species
# 6      6845885         Apis mellifera         Species
# 7      6971717      Apis nigrocincta         Species
```

The result from the above call to *col_downstream()* is a data.frame that gives a number of columns of different information.

IUCN status

There are a number of things a user can do once they have the correct taxonomic names. One thing a user can do is ask about the conservation status of a species (IUCN Red List of Threatened Species http://www.iucnredlist.org). We have provided a set of functions, *iucn_summary* and *iucn_status*, to search for species names, and extract the status information, respectively. Here, you can search for the panther and lynx.

```
ia <- iucn_summary(c("Panthera uncia", "Lynx lynx"))
iucn_status(ia)

# Panthera uncia    Lynx lynx
#           "EN"         "LC"
```

It turns out that the panther has a status of endangered (EN) and the lynx has a status of least concern (LC).

Search for available genes in GenBank

Another use case available in taxize deals with genetic sequences. taxize has three functions to interact with GenBank to search for available genes (*get_genes_avail*), download genes by GenBank ID (*get_genes*), and download genes via taxonomic name search, including retrieving a congeneric if the searched taxon does not exist in the database (*get_seqs*). In this example, one can search for gene sequences for *Umbra limi*.

```
out <- get_genes_avail(taxon_name = "Umbra limi",
                       seqrange = "1:2000",
                       getrelated = FALSE)
```

Then one can ask if 'RAG1' exists in any of the gene names.

```
out[grep("RAG1", out$genesavail, ignore.case = TRUE), -3]

#         spused  length    access_num         ids
# 413 Umbra limi     732      JX190826   394772608
# 427 Umbra limi     959      AY459526    45479841
# 434 Umbra limi    1631      AY380548    38858304
```

It turns out that there are 430 different unique records found. However, this doesn't mean that there are 430 different genes found as the API does not provide metadata to classify genes. You can use regular expressions (e.g., *grep*) to search for the gene of interest.

Matching species tables with different taxonomic resolution

Biologists often need to match different sets of data tied to species. For example, trait-based approaches are a promising tool in ecology[23]. One problem is that abundance data must be matched with trait databases such as the NCBI Taxonomy database[24]. These two data tables may contain species information on different taxonomic levels and data might have to be aggregated to a joint taxonomic level, so that the data can be merged. taxize can help in this data-cleaning step, providing a reproducible workflow.

A user can use the mentioned *classification*-function to retrieve the taxonomic hierarchy and then search the hierarchies up- and down-

wards for matches. Here is an example to match a species (A) with names of on different taxonomic levels (B1 & B2).

```
A <- "gammarus roeseli"
B1 <- "gammarus"
B2 <- "gammarus"
A_clas <- classification(A, db = 'ncbi')
B1_clas <- classification(B1, db = 'ncbi')
B2_clas <- classification(B2, db = 'ncbi')
A_clas[[1]]$Rank[tolower(A_clas[[1]]$ScientificName) %in% B1]

# [1] "genus"

A_clas[[1]]$Rank[tolower(A_clas[[1]]$ScientificName) %in% B2]

# [1] "family"
```

If one finds a direct match (here *Gammarus roeseli*), they will be lucky. However, Gammaridae can also be matched with *Gammarus roeseli*, but on a lower taxonomic level. A more comprehensive and realistic example (matching a trait table with an abundance table) is given in Appendix B.

Aggregating data to a specific taxonomic rank

In biology, one can ask questions at varying taxonomic levels. This use case is easily handled in taxize. A function called *tax_agg* will aggregate community data to a specific taxonomic level. In this example, one can take the data for three species and aggregate them to family level. Again one can specify whether they want to use data from ITIS or NCBI. The rows in the *data.frame* are different communities.

```
data(dune, package = 'vegan')
df <- dune[ , c(1,3:4)]
colnames(df) <- c("Bellis perennis","Juncus bufonius",
                  "Juncus articulatus")
head(df)

#   Bellis perennis   Juncus bufonius   Juncus articulatus
# 2               3                 0                   0
# 13              0                 3                   0
# 4               2                 0                   0
# 16              0                 0                   3
# 6               0                 0                   0
# 1               0                 0                   0
```

```
agg <- tax_agg(df, rank = 'family', db = 'ncbi')
agg

#
# Aggregated community data
#
# Level of Aggregation: FAMILY
# No. taxa before aggregation: 3
# No. taxa after aggregation: 2
# No. taxa not found: 0
```

```
head(agg$x)

#     Asteraceae     Juncaceae
# 2             3             0
# 13            0             3
# 4             2             0
# 16            0             3
# 6             0             0
# 1             0             0
```

The two *Juncus* species are aggregated to the family Juncaceae and their abundances are summed. There was only a single species in the family Asteraceae, so the data for *Bellis perennis* are carried over.

Conclusions

Taxonomic information is increasingly sought by biologists as we take phylogenetic and taxonomic approaches to science. Taxonomic data are becoming more widely available on the web, yet scientists require programmatic access to this data for developing reproducible workflows. taxize was created to bridge this gap - to bring taxonomic data on the web into R, where the data can be easily manipulated, visualized, and analyzed in a reproducible workflow.

We have outlined a suite of use cases in taxize that will likely fit real use cases for many biologists. Of course we have not thought of all possible use cases, so we hope that the biology community can give us feedback on what use cases they want to see available in taxize. One thing we could change in the future is to make functions that fit use cases, and then allow users to select the data source as a parameter in the function. This could possibly make the user interface easier to understand.

taxize is currently under development and will be for some time given the large number of data sources knitted together in the package, and the fact that APIs for each data source can change, requiring changes in taxize code. Contributions to taxize are strongly encouraged, and can be easily done using GitHub here https://github.com/ropensci/taxize_. We hope taxize will be taken up by the community and developed collaboratively, making it progressively better through time as new use cases arise, bug reports are squashed, and contributions are merged.

Author contributions

SC and ES equally contributed to the software discussed in this paper, and contributed equally to writing of the manuscript.

Competing interests

No competing interests were disclosed.

Grant information

The author(s) declared that no grants were involved in supporting this work.

Acknowledgements

The taxize package is part of the rOpenSci project http://ropensci.org/. We thank Carl Boettiger, Karthik Ram, Owen Jones, Naim Matasci, and Ralf Schäfer for comments on previous versions of this manuscript. We thank all API maintainers for their work making their databases open to the public.

References

1. Benton MJ: **Stems, nodes, crown clades, and rank-free lists: is linnaeus dead?** *Biol Rev Camb Philos Soc.* 2000; **75**(4): 633–648.
PubMed Abstract | Publisher Full Text
2. Weiser MD, Enquist BJ, Boyle B, *et al.*: **Latitudinal patterns of range size and species richness of new world woody plants**. *Global Ecology and Biogeography.* 2007; **16**(5): 679–688.
Publisher Full Text
3. Boyle B, Hopkins N, Lu Z, *et al.*: **The taxonomic name resolution service: an online tool for automated standardization of plant names.** *BMC Bioinformatics.* 2013; **14**(1): 16.
PubMed Abstract | Publisher Full Text | Free Full Text
4. Stodden VC: **Reproducible research: Addressing the need for data and code sharing in computational science.** *Comput Sci Eng.* 2010; **12**(5): 8–13.
Publisher Full Text
5. Zimmer C: **A sharp rise in retractions prompts calls for reform.** *New York Times.* 2012.
Reference Source
6. Gentleman RC, Carey VJ, Bates DM, *et al.*: **Bioconductor: open software development for computational biology and bioinformatics.** *Genome Biol.* 2004; **5**(10): R80.
PubMed Abstract | Publisher Full Text | Free Full Text
7. Federhen S: **The ncbi taxonomy database.** *Nucleic Acids Res.* 2012; **40** (Database issue): D136–D143.
PubMed Abstract | Publisher Full Text | Free Full Text
8. Webb CO, Donoghue MJ: **Phylomatic: tree assembly for applied phylogenetics.** *Mol Ecol Notes.* 2005; **5**(1): 181–183.
Publisher Full Text
9. Carvalho GH, Cianciaruso MV, Batalha MA: **Plantminer: a web tool for checking and gathering plant species taxonomic information.** *Environmental Modelling & Software.* 2010; **25**(6): 815–816.
Publisher Full Text
10. Chamberlain S, Ushey K: **rsnps: Interface to SNP data on the web.** R package version 0.0.4 2013.
Reference Source
11. Winter D: **rentrez: Entrez in R**, R package version 0.2.1 2013.
Reference Source
12. Lefeuvre P: **BoSSA: a Bunch of Structure and Sequence Analysis,** R package version 1.2 2010.
Reference Source
13. Paradis E, Claude J, Strimmer K: **APE: analyses of phylogenetics and evolution in R language.** *Bioinformatics.* 2004; **20**(2): 289–290.
PubMed Abstract | Publisher Full Text
14. Wickham H: **httr: Tools for working with URLs and HTTP.** R package version 0.2 2012.
Reference Source
15. Lang DT: **RCurl: General network (HTTP/FTP/...) client interface for R.** R package version 1.95–4.1 2013.
Reference Source
16. Lang DT: **XML: Tools for parsing and generating XML within R and S-Plus.** R package version 3.95–0.2 2013.
Reference Source
17. Lang DT: **RJSONIO: Serialize R objects to JSON, JavaScript Object Notation.** R package version 1.0–3 2013.
Reference Source
18. Wickham H: **stringr: Make it easier to work with strings.** R package version 0.6.2 2012.
Reference Source
19. Wickham H: **The split-apply-combine strategy for data analysis.** *J Stat Softw.* 2011; **40**(1): 1–29.
Reference Source
20. Webb CO, Ackerly DD, McPeek MA, *et al.*: **Phylogenies and community ecology.** *Annu Rev Ecol Syst.* 2002; **33**: 475–505.
Publisher Full Text
21. Rafferty NE, Ives AR: **Phylogenetic trait-based analyses of ecological networks.** *Ecology.* 2013.
Publisher Full Text
22. Poff NL, Olden JD, Vieira NKM, *et al.*: **Functional trait niches of north american lotic insects: traits-based ecological applications in light of phylogenetic relationships.** *Journal of the North American Benthological Society.* 2006; **25**(4): 730–755.
Publisher Full Text
23. Statzner B, Bêche LA: **Can biological invertebrate traits resolve effects of multiple stressors on running water ecosystems?** *Freshw Biol.* 2010; **55**(Supplement s1): 80–119.
Publisher Full Text
24. Usseglio-Polatera P, Bournaud M, Richoux P, *et al.*: **Biological and ecological traits of benthic freshwater macroinvertebrates: relationships and definition of groups with similar traits.** *Freshw Biol.* 2000; **43**(2): 175–205.
Publisher Full Text
25. Xie Y: **Dynamic Documents with R and knitr.** Chapman and Hall/CRC, 2013.
Reference Source
26. Baird DJ, Baker CJ, Brua RB, *et al.*: **Toward a knowledge infrastructure for traits-based ecological risk assessment.** *Integr Environ Assess Manag.* 2011; **7**(2): 209–215.
PubMed Abstract | Publisher Full Text
27. Kleyer M, Dray S, Bello F, *et al.*: **Assessing species and community functional responses to environmental gradients: which multivariate methods?** *Journal of Vegetation Science.* 2012; **23**(5): 805–821.
Publisher Full Text
28. Cayuela L: **Taxonstand: Taxonomic standardization of plant species names.** R package version 1.0 2012.
Reference Source
29. Cayuela L, Granzow-de la Cerda I, Al-buquerque FS, *et al.*: **taxonstand: An r package for species names standardisation in vegetation databases.** *Methods Ecol Evol.* 2012; **3**(6): 1078–1083.
Publisher Full Text

APPENDIX A. A COMPLETE REPRODUCIBLE WORKFLOW, FROM A SPECIES LIST TO A PHYLOGENY, AND DISTRIBUTION MAP.

If you aren't familiar with a complete workflow in R, it may be difficult to visualize the process. In R, everything is programmatic, so the whole workflow can be in one place, and be repeated whenever necessary. The following is a workflow for taxize, going from a species list to a phylogeny.

First, install taxize

```
install.packages("taxize")
```

Then load it into R

```
library(taxize)
```

Most of us will start out with a species list, something like the one below. Note that each of the names is spelled incorrectly.

```
splist <- c("Helanthus annuus", "Pinos contorta", "Collomia grandiflorra", "Rosa california",
"Mimulus bicolour", "Nicotiana glauca", "Maddia sativa")
```

There are many ways to resolve taxonomic names in taxize. Of course, the ideal name resolver will do the work behind the scenes for you so that you don't have to do things like fuzzy matching. There are a few services in taxize like this we can choose from: the Global Names Resolver service from EOL (see function *gnr_resolve*) and the Taxonomic Name Resolution Service from iPlant (see function *tnrs*). In this case let's use the function *tnrs*.

```
# The tnrs function accepts a vector of 1 or more
splist_tnrs <- tnrs(query = splist, getpost = "POST", source_ = "iPlant_TNRS")
# Remove some fields
(splist_tnrs <- splist_tnrs[, !names(splist_tnrs) %in% c("matchedName", "annotations",
"uri")])

#              submittedName             acceptedName          sourceId       score
# 5         Helanthus annuus         Helianthus annuus       iPlant_TNRS        0.98
# 1           Pinos contorta            Pinus contorta       iPlant_TNRS        0.96
# 7     Collomia grandiflorra       Collomia grandiflora       iPlant_TNRS        0.99
# 6          Rosa california          Rosa californica       iPlant_TNRS        0.99
# 4         Mimulus bicolour           Mimulus bicolor       iPlant_TNRS        0.98
# 3         Nicotiana glauca          Nicotiana glauca       iPlant_TNRS           1
# 2            Maddia sativa              Madia sativa       iPlant_TNRS        0.97

# Note the scores. They suggest that there were no perfect matches, but they
# were all very close, ranging from 0.77 to 0.99 (1 is the highest). Let's
# assume the names in the 'acceptedName' column are correct (and they should
# be).
# So here's our updated species list
(splist <- as.character(splist_tnrs$acceptedName))

# [1]   "Helianthus annuus"     "Pinus contorta"       "Collomia grandiflora"
# [4]   "Rosa californica"      "Mimulus bicolor"      "Nicotiana glauca"
# [7]   "Madia sativa"
```

Another thing we may want to do is collect common names for our taxa.

```
tsns <- get_tsn(searchterm = splist, searchtype = "sciname", verbose = FALSE)
comnames <- lapply(tsns, getcommonnamesfromtsn)
# Unfortunately, common names are not standardized like species names, so
# there are multiple common names for each taxon
sapply(comnames, length)
```

```
# [1] 3 3 3 3 3 3 3

# So let's just take the first common name for each species
comnames_vec <- do.call(c, lapply(comnames, function(x) as.character(x[1, "comname"])))
# And we can make a data.frame of our scientific and common names
(allnames <- data.frame(spname = splist, comname = comnames_vec))

#                  spname                          comname
# 1    Helianthus annuus                   common sunflower
# 2       Pinus contorta                    lodgepole pine
# 3  Collomia grandiflora              largeflowered collomia
# 4     Rosa californica                California wildrose
# 5      Mimulus bicolor    yellow and white monkeyflower
# 6     Nicotiana glauca                      tree tobacco
# 7         Madia sativa                     coast tarweed
```

Another common task is getting the taxonomic tree upstream from your study taxa. We often know what family or order our taxa are in, but it we often don't know the tribes, subclasses, and superfamilies. taxize provides many avenues to getting classifications. Two of them are accessible via a single function (*classification*): the Integrated Taxonomic Information System (ITIS) and National Center for Biotechnology Information (NCBI); and via the Catalogue of Life (see function *col_classification*):

```
# As we already have Taxonomic Serial Numbers from ITIS, let's just get
# classifications from ITIS. Note that we could use uBio instead.
class_list <- classification(tsns)
sapply(class_list, nrow)

# [1] 12 11 12 12 12 12 12

# And we can attach these names to our allnames data.frame
library(plyr)
gethiernames <- function(x) {
   temp <- x[, c("rankName", "taxonName")]
   values <- data.frame(t(temp[, 2]))
   names(values) <- temp[, 1]
   return(values)
}
class_df <- ldply(class_list, gethiernames)
allnames_df <- merge(allnames, class_df, by.x = "spname", by.y = "Species")
# Now that we have allnames_df, we can start to see some relationships among
# species simply by their shared taxonomic names
allnames_df[1:2, ]

#                spname       comname Kingdom             Subkingdom
# 1 Collomia grandiflora largeflowered collomia Plantae Viridaeplantae
# 2   Helianthus annuus       common sunflower Plantae Viridaeplantae
#   Infrakingdom     Division   Subdivision  Infradivision         Class
# 1 Streptophyta Tracheophyta Spermatophytina  Angiospermae Magnoliopsida
# 2 Streptophyta Tracheophyta Spermatophytina  Angiospermae Magnoliopsida
#   Superorder       Order        Family    Genus
# 1   Asteranae   Ericales Polemoniaceae  Collomia
# 2   Asteranae  Asterales    Asteraceae Helianthus

# Ah, so Abies and Bartlettia are in different infradivisions, but share
# taxonomic names above that point.
```

However, taxonomy can only get you so far. Shared ancestry can be reconstructed from molecular data, and phylogenies created. Phylomatic is a web service with an API that we can use to get a phylogeny.

```
# Fetch phylogeny from phylomatic
phylogeny <- phylomatic_tree(taxa = as.character(allnames$spname), taxnames = TRUE,
  get = "POST", informat = "newick", method = "phylomatic", storedtree = "R20120829",
  taxaformat = "slashpath", outformat = "newick", clean = "true", parallel = TRUE)
# Format teeth-labels
phylogeny$tip.label <- capwords(phylogeny$tip.label, onlyfirst = TRUE)
# plot phylogeny
plot(phylogeny)
```

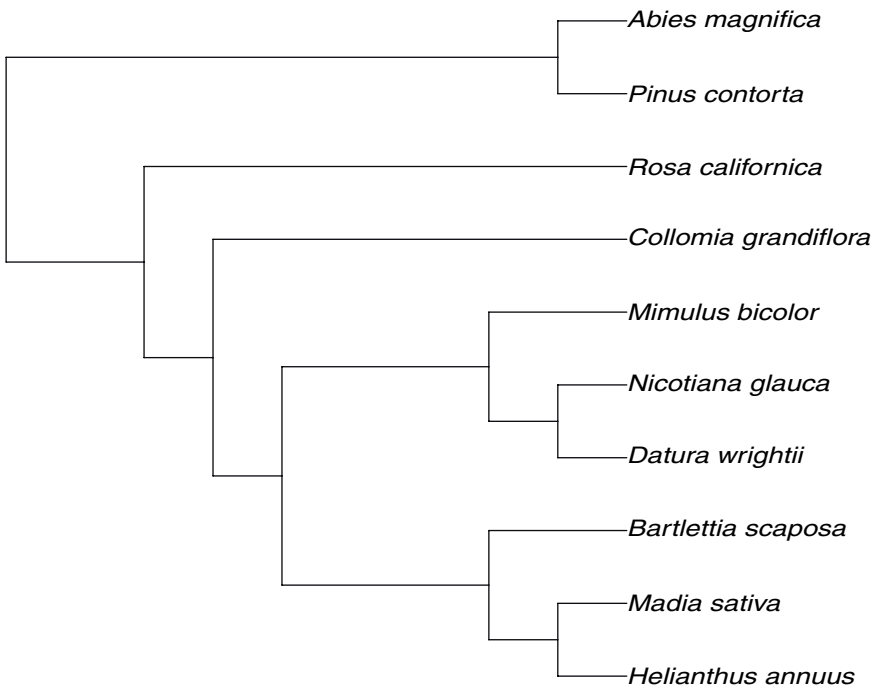

Figure A.1. A phylogeny.

Using the species list, with the corrected names, we can now search for occurrence data. The Global Biodiversity Information Facility (GBIF) has the largest collection of records data, and has a API that we can interact with programmatically from R. First, we need to install rgbif.

```
# Install rgbif from github.com
install.packages("devtools")
library(devtools)
install_github("rgbif", "ropensci")
```

Now we can search for occurrences for our species list and make a map.

```
library(rgbif)
library(ggplot2)
# get occurences
occurr_list <- occurrencelist_many(as.character(allnames$spname), coordinatestatus = TRUE,
    maxresults = 100, removeZeros = TRUE, fixnames = "changealltorig")
# Make a map
p <- gbifmap_list(occurr_list) + guides(col = guide_legend(title = "", nrow = 3,
   byrow = TRUE)) + theme(legend.position = "bottom", legend.key = element_blank()) +
```

```
    coord_equal()
p
```

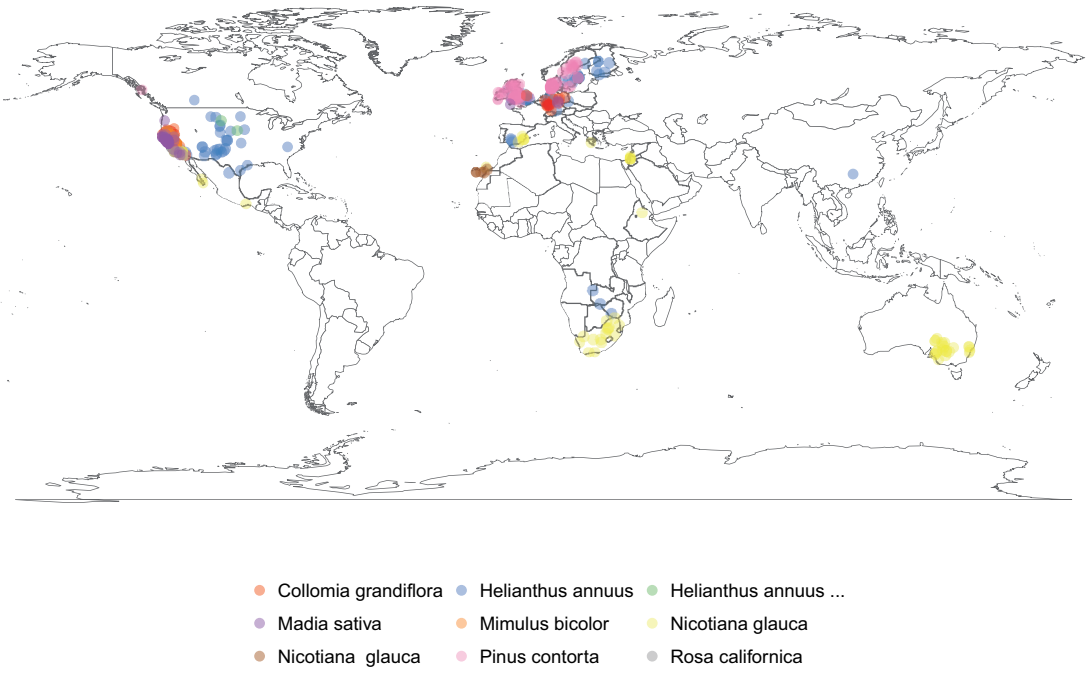

Figure A.2. A map.

APPENDIX B. MATCHING SPECIES TABLES WITH DIFFERENT TAXONOMIC RESOLUTION

Trait-based approaches are a promising tool in ecology. Unlike taxonomy-based methods, traits may not be constrained to biogeographic boundaries[27] and have potential to disentangle the effects of multiple stressors[24].

To analyse trait-composition abundance data must be matched with trait databases like[25]. However these two datatables may contain species information on different taxonomic levels and perhaps data must be aggregated to a joint taxonomic level.

taxize can help in this data-cleaning step, providing a reproducible workflow. Here we illustrate this on a small fictitious example.

Suppose we have fuzzy coded trait table with 2 traits with 3 respectively 2 modalities:

```
(traits <- read.table(header = TRUE, sep = ';', stringsAsFactors=FALSE,
                      text = 'taxon;T1M1;T1M2;T1M3;T2M1;T2M2
Gammarus sp.;0;0;3;1;3
Potamopyrgus antipodarum;1;0;3;1;3
Coenagrion sp.;3;0;1;3;1
Enallagma cyathigerum;0;3;1;0;3
Erythromma sp.;0;0;3;3;1
Baetis sp.;0;0;0;0;0
'))
```

	taxon	T1M1	T1M2	T1M3	T2M1	T2M2
1	Gammarus sp.	0	0	3	1	3
2	Potamopyrgus antipodarum	1	0	3	1	3
3	Coenagrion sp.	3	0	1	3	1
4	Enallagma cyathigerum	0	3	1	0	3
5	Erythromma sp.	0	0	3	3	1
6	Baetis sp.	0	0	0	0	0

And want to match this to a table with abundances:

```
(abundances <- read.table(header = TRUE, sep = ';', stringsAsFactors=FALSE,
                          text = 'taxon;abundance;sample
Gammarus roeseli;5;1
Gammarus roeseli;6;2
Gammarus tigrinus;7;1
Gammarus tigrinus;8;2
Coenagrionidae;10;1
Coenagrionidae;6;2
Potamopyrgus antipodarum;10;1
xxxxx;10;2
'))
```

	taxon	abundance	sample
1	Gammarus roeseli	5	1
2	Gammarus roeseli	6	2
3	Gammarus tigrinus	7	1
4	Gammarus tigrinus	8	2
5	Coenagrionidae	10	1
6	Coenagrionidae	6	2
7	Potamopyrgus antipodarum	10	1
8	xxxxx	10	2

First we do some basic data-cleaning and create a lookup-table, that will link taxa in trait table with the abundance table.

```
# first we remove ' sp.' from out trait table:
traits$taxon_cleaned <- tolower(gsub(" sp.", "", traits$taxon))
```

```
# since abundance tables can be very long with repeating taxa, we look only
# at unique taxon names This will be a lookup-table linking taxon names
# between both tables
lookup <- data.frame(taxon = tolower(unique(abundances$taxon)), stringsAsFactors = FALSE)
```

The we query the taxonomic hierarchy for both tables, this will be the backbone of this procedure:

```
library(taxize)
traits_classi <- classification(get_uid(traits$taxon_cleaned))
lookup_classi <- classification(get_uid(lookup$taxon))
```

First we look if we can find any direct matches between taxon names:

```
# first search for direct matches
direct <- match(lookup$taxon, traits$taxon_cleaned)
# and add the matched name to our lookup table
lookup$traits <- tolower(traits$taxon[direct])
lookup$match <- ifelse(!is.na(direct), "direct", NA)
lookup

                        taxon            traits        match
1            gammarus roeseli              <NA>         <NA>
2           gammarus tigrinus              <NA>         <NA>
3              coenagrionidae              <NA>         <NA>
4  potamopyrgus antipodarum potamopyrgus antipodarum direct
5                       xxxxx              <NA>         <NA>
```

We found a direct match - *potamopyrgus antipodarum* - so nothing to do here.
Next we look for species which are on a higher taxonomic resolution than our trait table. For these species we will take directly the trait-data since no better information is available.

```
# look for cases where taxonomic resolution in abundance data is higher than
# in trait data: here we take the trait-values for the lower resolution
for (i in which(is.na(lookup$traits))) {
     if (is.data.frame(lookup_classi[[i]])) {
         matches <- tolower(lookup_classi[[i]]$ScientificName) %in% traits$taxon_cleaned
         if (any(matches)) {
             lookup$traits[i] <- tolower(lookup_classi[[i]]$ScientificName[matches])
             lookup$match[i] <- lookup_classi[[i]]$Rank[matches]
     }
    }
  }
lookup

                      taxon                   traits   match
1          gammarus roeseli                 gammarus   genus
2         gammarus tigrinus                 gammarus   genus
3            coenagrionidae                     <NA>    <NA>
4  potamopyrgus antipodarum potamopyrgus antipodarum direct
5                     xxxxx                     <NA>    <NA>
```

So our abundance data has two *Gammarus* species, however trait data is only on genus level.
The next step is to search for species were we have to aggregate trait-data, since our abundance data is on a lower taxonomic level. We are walking the taxonomic latter for the species in our trait-data upwards and search for matches with out abundance data. Since we'll have many taxa in the trait-data belonging to one taxon, we'll take the median modality scores as an approximation. Of course also other methods may be used here, e.g. weighting by genetic distance.

```
# look for cases taxonomic resolution in abundance data is lower than in
# trait data, here we need to aggregate the trait-values (eg. median value
# for modality)
for (i in which(is.na(lookup$traits))) {
    # find matches
    agg <- sapply(traits_classi, function(x) any(tolower(x$ScientificName) %in%
        lookup$taxon[i]))
    if (sum(agg) > 1) {
        # add taxon as aggregate to trait-table
        traits <- rbind(traits, c(paste0(lookup$taxon[i], "-aggregated"), apply(traits[agg,
            2:6], 2, median), paste0(lookup$taxon[i], "-aggregated")))
    # fill lookup table
    lookup$traits[i] <- paste0(lookup$taxon[i], "-aggregated")
    lookup$match[i] <- "aggregated"
   }
}
lookup

#                       taxon                       traits       match
# 1          gammarus roeseli                    gammarus       genus
# 2         gammarus tigrinus                    gammarus       genus
# 3            coenagrionidae   coenagrionidae-aggregated   aggregated
# 4  potamopyrgus antipodarum    potamopyrgus antipodarum       direct
# 5                     xxxxx                         <NA>        <NA>
```

Finally we have only one taxon left - clearly an error. We remove this from our dataset:

```
abundances <- abundances[!abundances$taxon == lookup$taxon[is.na(lookup$traits)],
    ]
```

Now we can create *species x sites* and *traits x species* matrices, which could be plugged into methods to analyse trait responses[28].

```
# species (as matched with trait table) by site matrix
abundances$traits_taxa <- lookup$traits[match(tolower(abundances$taxon), lookup$taxon)]
library(reshape2)
# reshape data to long format and name rows by samples
L <- dcast(abundances, sample ~ traits_taxa, fun.aggregate = sum, value.var = "abundance")
rownames(L) <- L$sample
L$sample <- NULL
L

#  coenagrionidae-aggregated    gammarus    potamopyrgus antipodarum
# 1                        10          12                          10
# 2                         6          14                           0

# traits by species matrix
Q <- traits[, 2:7][match(names(L), traits$taxon_cleaned), ]
rownames(Q) <- Q$taxon_cleaned
Q$taxon_cleaned <- NULL
Q

#                             T1M1    T1M2    T1M3    T2M1    T2M2
# coenagrionidae-aggregated      0       0       1       3       1
# gammarus                       0       0       3       1       3
# potamopyrgus antipodarum       1       0       3       1       3

# check
all(rownames(Q) == colnames(L))

# [1] TRUE
```

This is just an example how taxonomic APIs (via taxize) could be used to search for matches up- and downwards the taxonomic ladder. We are looking forward to integrate the freshwaterecology.info database www.freshwaterecology.info into taxize, which will facilitate trait-based analyses in R.

APPENDIX C. INSTALLATION OF THE DEVELOPMENT VERSION OF TAXIZE AND API KEYS

Installing and using the development version of taxize

Stable versions of taxize are available on the Comprehensive R Archive Network (CRAN) by the following process:

```
install.packages("taxize")
library(taxize)
```

Development versions of taxize are available at Github at this link https://github.com/ropensci/taxize_, where the code-base is actively developed. This is also a good place to report bugs, submit feature requests, etc. on the Issues page https://github.com/ropensci/taxize_/issues.

The process of installing is a little bit more involved than from CRAN, but still quite easy using the package devtools http://cran.r-project.org/web/packages/devtools/index.html. If you don't have it yet, you can install it from CRAN:

```
install.packages("devtools")
```

You also need to install development tools if you haven't already:

- On Windows, download and install Rtools: http://cran.r-project.org/bin/windows/Rtools/. This is not an R package.
- On Mac, make sure you have either XCode (free, available in the app store) or the "Command Line Tools for Xcode" (needs a free apple id, available from http://developer.apple.com/downloads)

You can check you have everything installed and working by running this code:

```
library(devtools)
has_devel( )
```

Which should return 'TRUE'. Once that is taken care of, install taxize from Github.

```
install_github("taxize_", "ropensci")
```

Then load taxize into R.

```
library(taxize)
```

See an introduction to devtools here http://adv-r.had.co.nz/Philosophy.html.

API keys

Some of the data sources we provide access to in taxize require authentication through API (Application Program- ming Interface) keys. Navigate to your *.Rprofile* file, which should be

```
open .Rprofile
```

Then write in your API key to that file and save. Let's say we are writing a key for uBio. Put an entry in your .Rprofile file with a key of *uBioApi* and a value of your API key in quotes. You'll also need to restart R after you save your .Rprofile file.

```
# uBio API key
options(uBioApi = "youralphanumerickey")
```

When you use the taxize package, the function *ubio_namebank()* will look for that key and use it inthe API call. If the key is not found in your .Rprofile file the function will fail and tell you the key could not be found.

Alternatively, you can pass in the key in the function call like *ubio_namebank(searchName = 'elephant', sci = 1, vern = 0, keyCode=yourapikey)*

Functions in taxize that require API keys look for key values like *uBioApi* in your .Rprofile file. Therefore, unless you are passing your API key in the function call, save your keys in your .Rprofile file with the following key names (and their associated function names):

- *uBio*: ubio_namebank
- *EOLApi*: eol_dataobjects, eol_hierarchy, eol_pages, eol_ping, eol_search
- *tropicoskey*: tp_acceptednames, tp_namedistributions, tp_namereferences, tp_summary, tp_synonyms
- *pmkey*: plantminer.

Current Referee Status:

Referee Responses for Version 2

Gavin Simpson
Department of Biology, University of Regina, Regina, SK, Canada

Approved: 01 November 2013

Referee Report: 01 November 2013
The author's have addressed my comments on the original version of their manuscript. The issues I pointed to regarding naming conventions were not intended to be addressed now but in a future version of the package. The new appendix is a good addition to the manuscript, although I don't fully follow why "open .Rprofile" is highlighted as code?

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Competing Interests: No competing interests were disclosed.

1 Comment

Author Response

Scott Chamberlain, Simon Fraser University, Canada
Posted: 05 Nov 2013

Thanks for the feedback Gavin. Good point that the line "open .Rprofile" should not be highlighted as code. I will see if we can fix that. - Scott
Competing Interests: No competing interests were disclosed.

Will Pearse
College of Biological Sciences, University of Minnesota, Minneapolis, MN, USA

Approved: 29 October 2013

Referee Report: 29 October 2013
The authors have addressed all my concerns, and I think it's perfectly reasonable to address naming conventions in the next release of the package. The third appendix is a particularly nice addition; if the authors intend to alter the manuscript when they next update the package, the section entitled 'API Keys' may contain a minor typo/area that's not perfectly clear (see below) but I was still able to follow the appendix to alter my .Rprofile.

"Navigate to your .Rprofile file, which should be
open .Rprofile"

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Competing Interests: No competing interests were disclosed.

1 Comment

Author Response

Scott Chamberlain, Simon Fraser University, Canada
Posted: 05 Nov 2013

Thanks for your comments Will. Gavin mentioned something about the open .Rprofile as well, and we will fix anything there. - Scott
Competing Interests: No competing interests were disclosed.

Ethan White
Department of Biology, Utah State University, Logan, UT, USA

Approved: 28 October 2013

Referee Report: 28 October 2013
The current version of the manuscript addresses all of the recommendations in my previous review. I look forward to seeing the in-progress improvements in the software in its next release.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Competing Interests: No competing interests were disclosed.

1 Comment

Author Response

Scott Chamberlain, Simon Fraser University, Canada
Posted: 05 Nov 2013

Thanks for your constructive comments Ethan. - Scott
Competing Interests: No competing interests were disclosed.

Referee Responses for Version 1

Ethan White
Department of Biology, Utah State University, Logan, UT, USA

Approved: 24 September 2013

Referee Report: 24 September 2013
This software paper describes an R package that provides an integrated R interface for the APIs of over a dozen taxonomically related web services. This is a valuable contribution because it will save researchers time and energy (for those capable of wrapping the APIs themselves), and will allow scientists who lack the technical knowledge to interact with web services themselves to use this data from R. In addition, some of the functions combine existing APIs in useful ways.

The software was developed using version control, on a public development site (https://github.com/ropensci/taxize_), and using a bug tracker. The code is well modularized and includes an extensive test suite. This level of good software practice is notable for scientific software and is indicative of well built and maintained code. It also has a clearly declared CC0 license making it easy for others to use and build on the software.

The software installed easily in R using the standard approach and generally works as expected based on the examples in the paper and on the project's website.

My one major suggestion is to reinforce what the authors' have already suggested in the Conclusions, that it would be an improvement to move to a design that focuses on having a single top-level function for each type of task that the library handles with different data sources being selected using a parameter. This would allow the users to benefit maximally from one of the stronger aspects of this library, which is that it combines access to large numbers of data sources, by making it more of an integrated system and less of a collected set of API wrappers.

Minor issues:

1. The code snippets are images rather than text. This is probably a limitation of the publishing platform, but it does make it more difficult to learn about the software by executing the code snippets.

2. The following two sentences seem rather tangential to the paper and could be removed: *"Science workflows can now easily incorporate text, code, and images in a single executable document. Reproducible documents should become mainstream in biology to avoid mistakes, and make collaboration easier."*

3. Given that many of the target users of this package will not be particularly familiar with web services and APIs, I would recommend adding another sentence or two to the paragraph on

authentication so that readers understand why this is required (i.e., most readers won't understand "*users that abuse the API*") and what it really is (i.e., an individual login of sorts, similar to a username/password).

4. The section on "Aggregating data to a specific taxonomic rank" refers to an example, but none appears to be present.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Competing Interests: No competing interests were disclosed.

1 Comment

Author Response

Scott Chamberlain, Simon Fraser University, Canada
Posted: 14 Oct 2013

We appreciate Dr. White's comments on our manuscript.

We removed the sentences "*Science workflows can now easily incorporate text, code, and images in a single executable document. Reproducible documents should become mainstream in biology to avoid mistakes, and make collaboration easier*."

In response to this reviewer's comment about clarification on APIs and authentication, and Dr. Pearse's comments on the same issue, we have added a new appendix (Appendix C) that explains how to use API keys and install the development version of taxize.

This reviewer commented that the section on Aggregating data to a specific taxonomic rank referred to an example, but none appeared to be present. The example is now in the paper.
Competing Interests: No competing interests were disclosed.

Gavin Simpson
Department of Biology, University of Regina, Regina, SK, Canada

Approved: 23 September 2013

Referee Report: 23 September 2013
Chamberlain and Szöcs present the taxize R package, a set of functions that provides interfaces to several web tools and databases, and simplifies the process of checking, updating, correcting and manipulating taxon names for researchers working with ecological/biological data. A key feature that is repeated throughout is the need for reproducibility of science workflows and taxize provides a means to achieve this within the R software ecosystem for taxonomic search.

The manuscript is well-written and nicely presented, with a good balance of descriptive text and discourse and practical illustration of package usage. A number of examples illustrate the scope of the package, something that is fully expanded upon in the two appendices, which are a welcome addition to the paper.

As to the package, I am not overly fond of long function names; the authors should consider dropping the data source abbreviations from the function names in a future update/revision of the package. Likewise there is some inconsistency in the naming conventions used. For example there is the 'tpl_search()' function to search The Plant List, but the equivalent function to search uBio is 'ubio_namebank()'. Whilst this may reflect specific aspects of terminology in use at the respective data stores, it does not help the user gain familiarity with the package by having them remember inconsistent function names.

One advantage of taxize is that it draws together a rich selection of data stores to query. A further suggestion for a future update would be to add generic function names, that apply to a database connection/information object. The latter would describe the resource the user wants to search and any other required information, such as the API key, etc., for example:

```
foo <- taxizeDB(what = "uBio", key = "1646546164694")
```

The user function to search would then be 'search(foo, "Abies")'. Similar generically named functions would provide the primary user-interface, thus promoting a more consistent toolbox at the R level. This will become increasingly relevant as the scope of taxize increases through the addition of new data stores that the package can access.

In terms of presentation in the paper, I really don't like the way the R code inputs merge with the R outputs. I know the author of Knitr doesn't like the demarcation of output being polluted by the R prompt, but I do find it difficult parsing the inputs/outputs you show because often there is no space between them and users not familiar with R will have greater difficulties than I. Consider adding in more conventional indications of R outputs, or physically separate input from output by breaking up the chunks of code to have whitespace between the grey-background chunks. Related, in one location I noticed something amiss with the layout; in the first code block at the top of page 5, the printed output looks wrong here. I would expect the attributes to print on their own line and the data in the attribute to also be on its own separate line.

Note also, the inconsistency in the naming of the output object columns. For example, in the two code chunks shown in column 1 of page 4, the first block has an object printed with column names 'matched_name' and 'data_source_title', whilst camelCase is used in the outputs shown in the second block. As the package is revised and developed, consider this and other aspects of providing a consistent presentation to the user.

I was a little confused about the example in the section Resolve Taxonomic Names on page 4. Should the taxon name be "*Helianthus annuus*" or "*Helianthus annus*"? In the 'mynames' definition you include ' *Helianthus annuus*' in the character vector but the output shown suggests that the submitted name was ' *Helianthus annus*' (1 "u") in rows with rownames 9 and 10 in the output shown.

Other than that there were the following minor observations:

1. Abstract: replace "easy" with "simple" in "...*fashion that's easy*...", and move the details about availability and the URI to the end of the sentence.

2. Page 2, Column 1, Paragraph 2: You have "*In addition, there is no one authoritative taxonomic names source*...", which is a little clumsy to read. How about "*In addition, there is no one authoritative source of taxonomic names*..."?
3. Pg 2, C1, P2-3: The abbreviated data sources are presented first (in paragraph 2) and subsequently defined (in para 3). Restructure this so that the abbreviated forms are explained upon first usage.
4. Pg 2, C2, P2: Most R packages are "in development" so I would drop the qualifier and reword the opening sentence of the paragraph.
5. Pg 2, C2, P6: Change "*and more can easily be added*" to "*and more can be easily added*" seems to flow better?
6. Pg 5, paragraph above Figure 1: You refer to converting the object to an **ape** *phylo* object and then repeat essentially the same information in the next sentence. Remove the repetition.
7. Pg 6, C1: The header may be better as "*Which taxa are children of the taxon of interest*".
8. Pg 6: In the section "IUCN status", the term "we" is used to refer to both the authors and the user. This is confusing. Reserve "we" for reference to the authors and use something else ("a user" perhaps) for the other instances. Check this throughout the entire manuscript.
9. Pg 6, C2: in the paragraph immediately below the 'grep()' for "RAG1", two consecutive sentences begin with "However".
10. Pg 7: The first sentence of "Aggregating data...." reads "*In biology, one can asks questions*...". It should be "*one ask*s" or "*one can ask*".
11. Pg 7, Conclusions: The first sentence reads *"information is increasingly sought out by biologists*". I would drop "out" as "sought" is sufficient on its own.
12. Appendices: Should the two figures in the Appendices have a different reference to differentiate them from Figure 1 in the main body of the paper? As it stands, the paper has two Figure 1s, one on page 5 and a second on page 12 in the Appendix.
13. On Appendix Figure 2: The individual points are a little large. Consider reducing the plotting character size. I appreciate the effect you were going for with the transparency indicating density of observation through overplotting, but the effect is weakened by the size of the individual points.
14. Should the phylogenetic trees have some scale to them? I presume the height of the stems is an indication of phylogenetic distance but the figure is hard to calibrate without an associated scale. A quick look at Paradis (2012) Analysis of Phylogenetics and Evolution with R would suggest however that a scale is not consistently applied to these trees. I am happy to be guided by the authors as they will be more familiar with the conventions than I.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Competing Interests: No competing interests were disclosed.

1 Comment

Author Response

Scott Chamberlain, Simon Fraser University, Canada
Posted: 14 Oct 2013

We appreciate Dr. Simpson's very thorough comments on our manuscript. The following are responses to Dr. Simpson's comments:

- ...there is some inconsistency in the naming conventions used. For example there is the 'tpl_search()' function to search The Plant List, but the equivalent function to search uBio is 'ubio_namebank()'. Whilst this may reflect specific aspects of terminology in use at the respective data stores, it does not help the user gain familiarity with the package by having them remember inconsistent function names.

We agree that we should definitely improve naming conventions for functions. However, we think it's better to change the function names as needed in an upcoming version of the software after we have had time work on the problem.

- Consider adding in more conventional indications of R outputs, or physically separate input from output by breaking up the chunks of code to have whitespace between the grey-background chunks.

We have used comments (pound signs) for the results of function calls within each code block to indicate output as separate from code input. This way users can copy/paste code directly into R to try it out.

- in one location I noticed something amiss with the layout; in the first code block at the top of page 5, the printed output looks wrong here. I would expect the attributes to print on their own line and the data in the attribute to also be on its own separate line.

This was a problem with the typesetting, and we have fixed it.

- the inconsistency in the naming of the output object columns. For example, in the two code chunks shown in column 1 of page 4, the first block has an object printed with column names 'matched_name' and 'data_source_title', whilst camelCase is used in the outputs shown in the second block.

We agree that we should definitely improve naming conventions for object columns. However, we think it's better to change the column names as needed in an upcoming version of the software after we have had time work on the problem.

- I was a little confused about the example in the section Resolve Taxonomic Names on page 4. Should the taxon name be "Helianthus annuus" or "Helianthus annus"? In the 'mynames' definition you include 'Helianthus annuus' in the character vector but the output shown suggests that the submitted name was 'Helianthus annus' (1 "u") in rows with rownames 9 and 10 in the output shown.

Fixed.

- Abstract: replace "easy" with "simple" in "...fashion that's easy...", and move the details about availability and the URI to the end of the sentence.

Fixed.

- Page 2, Column 1, Paragraph 2: You have "In addition, there is no one authoritative taxonomic names source...", which is a little clumsy to read. How about "In addition, there is no one authoritative source of taxonomic names..."?

Changed.

- Pg 2, C1, P2-3: The abbreviated data sources are presented first (in paragraph 2) and subsequently defined (in para 3). Restructure this so that the abbreviated forms are explained upon first usage.

Changed.

- Pg 2, C2, P2: Most R packages are "in development" so I would drop the qualifier and reword the opening sentence of the paragraph.

Changed.

- Pg 2, C2, P6: Change "and more can easily be added" to "and more can be easily added" seems to flow better?

Changed.

- Pg 5, paragraph above Figure 1: You refer to converting the object to an ape phylo object and then repeat essentially the same information in the next sentence. Remove the repetition.

Removed.

- Pg 6, C1: The header may be better as "Which taxa are children of the taxon of interest".

Changed.

- Pg 6: In the section "IUCN status", the term "we" is used to refer to both the authors and the user. This is confusing. Reserve "we" for reference to the authors and use something else ("a user" perhaps) for the other instances. Check this throughout the entire manuscript.

Fixed.

- Pg 6, C2: in the paragraph immediately below the 'grep()' for "RAG1", two consecutive sentences begin with "However".

Changed.

- Pg 7: The first sentence of "Aggregating data...." reads "In biology, one can asks questions...". It should be "one asks" or "one can ask"

Changed.

- Pg 7, Conclusions: The first sentence reads "information is increasingly sought out by biologists". I would drop "out" as "sought" is sufficient on its own.

Changed.

- Appendices: Should the two figures in the Appendices have a different reference to differentiate them from Figure 1 in the main body of the paper? As it stands, the paper has two Figure 1s, one on page 5 and a second on page 12 in the Appendix.

Fixed.

- On Appendix Figure 2: The individual points are a little large. Consider reducing the plotting character size. I appreciate the effect you were going for with the transparency indicating density of observation through overplotting, but the effect is weakened by the size of the individual points.

Although we agree that the styling of the figure could be improved, we are going to leave it as is because it's not important for understanding the material.

- Should the phylogenetic trees have some scale to them? I presume the height of the stems is an indication of phylogenetic distance but the figure is hard to calibrate without an associated scale. A quick look at Paradis (2012) Analysis of Phylogenetics and Evolution with R would suggest however that a scale is not consistently applied to these trees. I am happy to be guided by the authors as they will be more familiar with the conventions than I.

A scale could be used for sure. However, our focus is on showing readers that they can get data which can be used to make a phylogeny, not on how to properly create and display a phylogeny. Thus, we are leaving the phylogeny as is without a scale.
Competing Interests: No competing interests were disclosed.

Will Pearse
College of Biological Sciences, University of Minnesota, Minneapolis, MN, USA

Approved: 19 September 2013

Referee Report: 19 September 2013
The software this article describes is well-written and of use to ecologists. The guide and appendices in this article give a good overview of the features of the package, and are well-written. The title and abstract are well-written.

My only serious comment would be that the authors refer to species' taxonomy as if it perfectly reflects species' phylogeny (e.g. the end of the first paragraph in the introduction and the first paragraph in the section "retrieve higher taxonomic names"). More often than not, taxonomy does reflect phylogeny, but a sentence clarifying this distinction somewhere might be helpful.

A few minor things:

- Table 1 - "National Center for Biotechnology Information Federhen" - the author's surname, but not the citation, is in the text; "searchbycommonname" and "searchbyscientificname" seem to be the only elements not in alphabetical order in the table.

- Phylomatic can now create phylogenies for mammals, not just angiosperms (e.g. "retrieve a phylogeny").
- Some sentences could perhaps be linked more neatly in the introduction (e.g. "*gives that name as unresolved. But Helianthus*", "*moved to the cloud. Therefore there is a need*", "*additional confusion. Last, name*"). Similarly, page 3 has some very short paragraphs; could the descriptions of *taxonstand* and The Plant List be moved to table 1? "Science workflows" or "scientific workflows" (fourth paragraph of introduction)? These are all very minor points!
- Perhaps an example of what to type in order to add API keys into the .Rprofile would be helpful, or maybe could just be added to the help file for "*ubio_namebank*" as this is the function named when it's mentioned.

I have read this submission. I believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Competing Interests: No competing interests were disclosed.

1 Comment

Author Response

Scott Chamberlain, Simon Fraser University, Canada
Posted: 14 Oct 2013

We appreciate Dr. Pearse's comments on our manuscript. We agree that species taxonomy does not equate to phylogenetic history, so we added the following sentence to the first paragraph: *"Although taxonomic classifications are human constructs created to understand the real phylogeny of life \cite{benton2000}, they are nonetheless essential to organize the vast diversity of organisms*." We fixed the citation in Table 1, and reordered the functions in the table so as to be alphabetical. Thanks for pointing out that Phylomatic now accepts mammals in addition to Angiosperm plants - we have adjusted the language accordingly.

We removed the description of Taxonstand and the Plantlist.org (and associated references) that this reviewer referred to as it wasn't necessary and improves reading.

This reviewer asked for a better explanation of how to use the API keys and the .Rprofile file. In response, we have added a new appendix (Appendix C) that explains using API keys and installing the development version of taxize.
Competing Interests: No competing interests were disclosed.