taxize - taxonomic search and retrieval in R

Scott Chamberlain[1] and Eduard Szöcs[2]
1 Biology Department, Simon Fraser University, Canada
2 Institute for Environmental Sciences, University Koblenz-Landau, Fortstr. 7, 76829 Landau, Germany
∗ E-mail: myrmecocystus@gmail.com

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 taxonomic data on the web 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 will facilitate open and reproducible science by allowing taxonomic data collection to be done in the open-source R platform.

Author Summary

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*). Biologists, whether studying organisms at the cell, organismal, or community level, can put their study taxa into taxonomic context, allowing them to know close and distant relatives, find relevant literature, and more.

The use of taxonomic names is, unfortunately, not straightforward. Taxonomic names often change due to name changes 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 [1, 2]. The scale of this study may be larger than most, but the problem persists for any size study. In addition, there is no one authoritative taxonomic names source for all taxa - although, there are taxon specific sources that many scientists use that study that taxon. Different sources (e.g., uBio, Tropicos, ITIS) may have different accepted names for the same taxon. For example, while the Integrated Taxonomic Information Service (ITIS) has *Helianthus x glaucus* as an accepted name, The Plant List `http://www.theplantlist.org` has that name as unresolved, but has *Helianthus glaucus* as an accepted name, while ITIS doesn't have the 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 the Universal Biological Indexer and Organizer (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. Yet, there are no universal codes for taxa across databases, leading 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 web interfaces (e.g., [3]). 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 lookup on a website, it can take quite a long time to enter each name, get data, and repeat for each species. Second, programatically getting taxonomic names solves the first problem by looping over a list of names. In addition, doing taxonomic searching, etc. is reproducible. With increasing reports of irreproducibility in science [4, 5], it is extremely important to make science workflows repeatable. Science workflows can now easily incorporate text, code, and images in a single executable document [6].

The R language is the dominant language used by biologists (reference), and now has over 4,500 packages on the Comprehensive R Archive Network (CRAN) and more than 2,500 packages on other repositories to extend R. R is great for manipulating, visualizing and fitting statistical models to data. However, the key missing piece in R is the ability to get data from the internet within R. Getting data from the web will be increasingly common as more and more data gets moved to the cloud. Increasingly, data is available from the web via application programming interfaces (API). These are bits of code that 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 strength 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. Whereas, if data are stored in an Excel file, for example, the file can only be opened in a few programs.

The goal of taxize, an R package in development, is to make all use cases having to do with retrieving and resolving taxonomic names easy and replicable. 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 do with a web API. Therefore, we hope moving from a web to R interface for taxonomic names will be relatively seamless (if one is already nominally familiar with R).

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

Why do we need taxize?

There are a large suite of applications developed around the problem of searching for, resolving, and getting higher taxonomy for species names. For example, Linnaeus [7] 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 [8], uBio tools [9], and iPlant's Taxonomic Name Resolution Service [10].

All of these tools provide ways to search for taxonomic names and resolve them in some cases. However, scientists ideally need a tool that can be used programmatically, and thereby facilitate reproducible research. The goal of taxize is to make it easy to create 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.

One could argue that a different programming language would have been better than R. For example, Python performs many actions faster than R, and Ruby plays really nicely in a browser, facilitating web applications. However, our goal with taxize is to create a product for researchers primarily, and the most common programming language for researchers, at least in the life sciences, is R. [11] gives a detailed discussion of advantages of R in computational biology.

Data sources and package details

taxize uses many data sources (Table 1), and more can easily be 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 [12], rentrez [13], BoSSA [14], and ape [15], so taxize will 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 so that they can better manage their servers, collect analytics, and shut down users that abuse the API. The services that do require an API key in taxize are: Encyclopedia of Life (EOL) [3], the Universal Biological Indexer and Organizer (uBio) [9], Tropicos [16], and Plantminer [17]. You can easily obtain an API key by visiting the website of each service (see Table 1 for links to each site). There are two ways of using your 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 API keys in your .Rprofile file. On a Mac this file is at */yourhomefolder/.Rprofile*; on a Windows machine at */yourhomefolder/.Rprofile*; and on Linux at */yourhomefolder/.Rprofile*. This is an hidden file, so open up this file in your terminal (e.g., *open .Rprofile*), and add the API key as an entry like *options(myapikey = 'your_alphanumeric_key')*. We recommend the second option as it simplifies function calls.

One available data source in taxize is The Plant List [18]. The connection in taxize is done via the *taxonstand* package [19] that solely interacts with that The Plant List. We provide a few convenience functions that wrap taxonstand into taxize.

taxize would not have been possible without the work of others. taxize uses httr [20] and RCurl [21] for doing calls to web APIs, XML [22] for parsing XML, RJSONIO [23] for parsing JSON, and stringr [**?**] and plyr [24] for manipulating data.

New data sources can be added; we may add the following sources: Wikispecies and The Tree of Life. A connection to `freshwaterecology.info` [25] (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 will be released. In addition, the authors may be contacted for further suggestions of data sources to be added.

Use cases

There are a variety of use cases for which taxize is ideally suited. We discuss here several ideal use cases for taxize at length. Moreover a complete reproducible workflow from a species list to a phylogeny and a distribution map can be found in the supplement.

First, install taxize

First, we must install taxize. There are two versions of taxize, a) a stable release that can be installed from the R package repository, CRAN, and b) from GitHub [26], where the code is developed.

Installing from CRAN or GitHub

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

```r
install.packages("devtools")
require(devtools)
install_github("taxize_", "ropensci")
```

Load taxize into your R session.

```r
library(taxize)
```

Resolve taxonomic names

This is a common task in biology. We often have a list of species names and we want to know if a) we have the most up to date names, b) our names are spelled correctly, and c) if we have common names, we likely need the scientific names. One way to resolve names is via the Global Names Resolver (GNR) service provided by the Encyclopedia of Life [8]. Here, we are searching for two misspelled names:

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

The correct spellings are *Helianthus annuus* and *Homo sapiens*. Another approach uses the Taxonomic Name Resolution Service via the Taxosaurus API [27] developed by iPLant and the Phylotastic organization. In this example, we provide a list of species names, some of which are misspelled, and we'll call the API with the *tnrs* function.

```r
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
7   Helianthus annuus   Helianthus annuus iPlant_TNRS  1.00
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
8     Rosa california    Rosa californica iPlant_TNRS  0.99
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 taxize making the decision. Both the EOL GNR and the TNRS services provide data from a variety of data sources. The user may trust a specific data source, 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, but since R is relatively inefficient in memory management, etc., we would rather offload this sort of computationally intensive task.

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) [28] and National Center for Biotechnology Information (NCBI) [29]. First, we'll search for two species, *Abies procera* and *Pinus contorta*.

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

[[1]]
        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

[[2]]
        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 doing *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
```

Interactive name selection

As mentioned 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 what 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 list 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"  "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 [30], ecological networks [31], functional trait ecology [32]. 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 [33]. 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.

```
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. The output from *phylomatic_tree()* is a *phylo* object, which can be used for plotting and phylogenetic analyses. Be aware that Phylomatic has certain limitations - refer to the paper describing Phylomatic [33] and the website `http://phylodiversity.net/phylomatic/`.

There are currently no resources for getting a phylogeny of animals simply from species names. However, a few projects are working on this problem, including the Open Tree of Life [34]. We will incorporate these resources when the appropriate APIs are available.

What taxa are the children of my taxon of interest?

If someone is not a taxonomic specialist on a particular taxon he likely does 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 [35] or Encyclopedia of Life [3] to get downstream names. However, taxize provides an easy way to programatically search for downstream taxa, both for the Catalogue of Life (CoL) [36] and the Integrated Taxonomic Information System [28]. 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 we can do once we have the correct taxonomic names. One thing we can do is ask about the conservation status of a species (IUCN Red List of Threatened Species [37]). We have provided a set of functions, *iucn_summary* and *iucn_status*, to search for species names, and extract the status information, respectively. Here, we 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 (LN).

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, we search for gene sequences for *Umbra limi*.

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

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

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

        spused length
413 Umbra limi    732
427 Umbra limi    959
434 Umbra limi   1631
                                                                             genesavail
413 isolate UlimA recombinase activating protein 1 (rag1) gene, exon 3 and partial cds
427           recombination-activating protein 1 (RAG1) gene, intron 2 and partial cds
434                        recombination-activating protein 1 (RAG1) gene, partial cds
    access_num       ids
413   JX190826 394772608
427   AY459526  45479841
434   AY380548  38858304
```

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

Matching species tables with different taxonomic resolution

Biologist often need to match different sets of data tied to species. For example, trait-based approaches are a promising tool in ecology [38]. One problem is that abundance data must be matched with trait databases like [39]. These two data tables may contain species information on different taxonomic levels and possibly data must be aggregated to a joint taxomic level, so that the data can be merged. taxize can help in this data-cleaning step, providing a reproducible workflow:

We can use the mentioned *classification*-function to retrieve the taxonomic hierarchy and then search the hierarchies up- and downwards for matches. Here is an example to match a species with names on three different taxonomic levels.

```
A <- "gammarus roeseli"

B1 <- "gammarus roeseli"
B2 <- "gammarus"
B3 <- "gammaridae"

A_clas <- classification(A, db = 'ncbi')
B1_clas <- classification(B1, db = 'ncbi')
B2_clas <- classification(B2, db = 'ncbi')
B3_clas <- classification(B3, db = 'ncbi')


B1[match(A, B1)]

[1] "gammarus roeseli"

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

[1] "genus"

A_clas[[1]]$Rank[tolower(A_clas[[1]]$ScientificName) %in% B3]

[1] "family"
```

If we find a direct match (here *Gammarus roeseli*), we are lucky. But we can also match Gammaridae 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 asks questions at varying taxonomic levels. One may perform analyses on different 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, we take data of 5 species and aggregate them to family level. Again we can specify if we want to use data from ITIS or NCBI.

```
data(dune, package = 'vegan')
df <- dune[ , 1:5]
colnames(df) <- c("Bellis perennis", "Empetrum nigrum", "Juncus bufonius",
                  "Juncus articulatus", "xxx")
head(df)
```

```
   Bellis perennis Empetrum nigrum Juncus bufonius Juncus articulatus xxx
2                3               0               0                  0   0
13               0               0               3                  0   0
4                2               0               0                  0   0
16               0               0               0                  3   0
6                0               0               0                  0   0
1                0               0               0                  0   0

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


Aggregated community data

Level of Aggregation: FAMILY
No. taxa before aggregation: 5
No. taxa after aggregation: 4
No. taxa not found: 1

head(agg$x)

   Asteraceae Ericaceae Juncaceae xxx
2           3         0         0   0
13          0         0         3   0
4           2         0         0   0
16          0         0         3   0
6           0         0         0   0
1           0         0         0   0
```

We see that the two Juncus species are aggregated to Juncaceae and their abundances are summed up. If a taxon is one lower taxonomic resolution then the querried or the taxon is not found in the database (like in the example taxon *xxx*) then these are not aggregated and returned as is.

Conclusions

Taxonomic information is increasingly sought out by biologists as we take phylogenetic and taxonomic approaches to science. Taxonomic data is quickly becoming available on the web, yet scientists require programmatic access to this data to create 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 of 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 kitted 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 [26]. We hope taxize will be taking up by the community and developed collaboratively, making it progressively better through time as new use cases arise, bug reports squashed, and contributions merged.

Acknowledgments

The taxize package is part of the rOpenSci project http://ropensci.org/. We thank X, Y, and Z for comments on previous versions of this manuscript. SAC is supported by CANPOLIN of Canada, grant number XXXXXX. We thank all API maintainers for their work making their databases open to the public.

References

1. Weiser MD, Enquist BJ, Boyle B, Killeen TJ, Jrgensen PM, et al. (2007) Latitudinal patterns of range size and species richness of new world woody plants. Global Ecology and Biogeography 16: 679-688.
2. Boyle B, Hopkins N, Lu Z, Raygoza Garay JA, Mozzherin D, et al. (2013) The taxonomic name resolution service: an online tool for automated standardization of plant names. BMC Bioinformatics 14: 16.
3. Encylopedia of Life (2013). Available: http://eol.org/. Accessed May 27 2013.
4. Stodden VC (2010) Reproducible research: Addressing the need for data and code sharing in computational science. Computing in Science & Engineering 12: 8–12.
5. Zimmer C (2012) A sharp rise in retractions prompts calls for reform. New York Times .
6. Xie Y (2013) Dynamic Documents with R and knitr. Chapman and Hall/CRC. URL http://yihui.name/knitr/.
7. Linnaeus (2013). Available: http://linnaeus.sourceforge.net/. Accessed May 27 2013.
8. Global Names Resolver (2013). Available: http://resolver.globalnames.org/. Accessed May 27 2013.
9. uBio (2013). Universal biological indexer and organizer. Available: http://www.ubio.org/index.php?pagename=sample_tools. Accessed May 27 2013.
10. TNRS (2013). Taxonomic name resolution service. Available: http://tnrs.iplantcollaborative.org/. Accessed May 27 2013.
11. Gentleman RC, Carey VJ, Bates DM, Bolstad B, Dettling M, et al. (2004) Bioconductor: open software development for computational biology and bioinformatics. Genome Biology 5: R80.
12. Chamberlain S, Ushey K (2013) rsnps: Interface to SNP data on the web. URL https://github.com/ropensci/rsnps. R package version 0.0.4.
13. Winter D (2013) rentrez: Entrez in R. URL https://github.com/ropensci/rentrez. R package version 0.2.1.
14. Lefeuvre P (2010) BoSSA: a Bunch of Structure and Sequence Analysis. URL http://CRAN.R-project.org/package=BoSSA. R package version 1.2.
15. Paradis E, Claude J, Strimmer K (2004) APE: analyses of phylogenetics and evolution in R language. Bioinformatics 20: 289-290.
16. Missouri Botanical Garden (2013). Tropicos.org. Available: http://www.tropicos.org/. Accessed May 27 2013.

17. Carvalho GH, Cianciaruso MV, Batalha MA (2010) Plantminer: a web tool for checking and gathering plant species taxonomic information. Environmental Modelling & Software 25: 815–816.

18. The Plant List (2013). A working list of all plant species. Available:`http://www.theplantlist.org`. Accessed May 27 2013.

19. Cayuela L, Granzow-de la Cerda , Albuquerque FS, Golicher DJ (2012) taxonstand: An r package for species names standardisation in vegetation databases. Methods in Ecology and Evolution 3: 1078–1083.

20. Wickham H (2012) httr: Tools for working with URLs and HTTP. URL `http://CRAN.R-project.org/package=httr`. R package version 0.2.

21. Lang DT (2013) RCurl: General network (HTTP/FTP/...) client interface for R. URL `http://CRAN.R-project.org/package=RCurl`. R package version 1.95-4.1.

22. Lang DT (2013) XML: Tools for parsing and generating XML within R and S-Plus. URL `http://CRAN.R-project.org/package=XML`. R package version 3.95-0.2.

23. Lang DT (2013) RJSONIO: Serialize R objects to JSON, JavaScript Object Notation. URL `http://CRAN.R-project.org/package=RJSONIO`. R package version 1.0-3.

24. Wickham H (2011) The split-apply-combine strategy for data analysis. Journal of Statistical Software 40: 1–29.

25. Schmidt-Kloiber A, Hering D (2013). www.freshwaterecology.info - the taxa and autecology database for freshwater organisms, version 5.0. Available: `www.freshwaterecology.info`.

26. taxize (2013). taxize on github. Available: `https://github.com/ropensci/taxize_`.

27. Taxosaurus (2013). The taxonomic thesaurus. Available: `http://taxosaurus.org/`. Accessed May 27 2013.

28. ITIS (2013). Integrated taxonomic information service. Available: `http://www.itis.gov/`. Accessed May 27 2013.

29. Federhen S (2012) The ncbi taxonomy database. Nucleic Acids Research 40: D136-D143.

30. Webb CO, Ackerly DD, McPeek MA, Donoghue MJ (2002) Phylogenies and community ecology. Annual Review of Ecology and Systematics : 475–505.

31. Rafferty NE, Ives AR (2013) Phylogenetic trait-based analyses of ecological networks. Ecology .

32. Poff NL, Olden JD, Vieira NK, Finn DS, Simmons MP, et al. (2006) Functional trait niches of north american lotic insects: traits-based ecological applications in light of phylogenetic relationships. Journal of the North American Benthological Society 25: 730–755.

33. Webb CO, Donoghue MJ (2005) Phylomatic: tree assembly for applied phylogenetics. Molecular Ecology Notes 5: 181–183.

34. Open Tree of Life (2013). Available: `http://blog.opentreeoflife.org/`. Accessed May 27 2013.

35. Wikispecies (2013). Available: `http://species.wikimedia.org/wiki/Main_Page`. Accessed May 27 2013.

36. Roskov Y, Kunze T, Paglinawan L, Orrell T, Nicolson D, et al. (2013). Catalogue of Life. Available: http://www.catalogueoflife.org/. Accessed May 27 2013.

37. IUCN (2013). Iucn red list of threatened species. Available: http://www.iucnredlist.org. Accessed May 27 2013.

38. Statzner B, Bêche L (2010) Can biological invertebrate traits resolve effects of multiple stressors on running water ecosystems? Freshwater Biology 55: 80119.

39. Usseglio-Polatera P, Bournaud M, Richoux P, Tachet H (2000) Biological and ecological traits of benthic freshwater macroinvertebrates: relationships and definition of groups with similar traits. Freshwater Biology 43: 175205.

40. Angiosperm Phylogeny Group (2013). Available: http://www.mobot.org/MOBOT/research/APweb/. Accessed May 27 2013.

41. Global Invasive Species Database (2013). Global invasive species database. Available: http://www.issg.org/database/welcome/. Accessed May 27 2013.

42. Global Names Index (2013). Available: http://gni.globalnames.org/. Accessed May 27 2013.

Figure Legends

Figure 1. A Phylogeny for the three species.

Tables

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 [40]
classification	Upstream classification	Various
col_children	Direct children	Catalogue of Life [36]
col_downstream	Downstream taxa to specified rank	Catalogue of Life [36]
eol_hierarchy	Upstream classification	Encyclopedia of Life [3]
eol_search	Search EOL taxon information	Encyclopedia of Life [3]
get_seqs	Get NCBI sequences	National Center for Biotechnology Information [29]
get_tsn	Get ITIS TSN	Integrated Taxonomic Information System [28]
get_uid	Get NCBI UID	National Center for Biotechnology Information [29]
searchbycommonname	Search ITIS by common name	Integrated Taxonomic Information System [28]
searchbyscientificname	Search ITIS by scientific name	Integrated Taxonomic Information System [28]
gisd_isinvasive	Inasiveness status	Global Invasive Species Database [41]
gni_parse	Parse scientific names into components	Global Names Index [3, 42]
gni_search	Search EOL's global names index	Global Names Index [3, 42]
gnr_resolve	Resolve names using EOL's global names index	Global Names Resolver [3, 8]
itis_downstream	Downstream taxa to specified rank	Integrated Taxonomic Information System [28]
iucn_status	IUCN status	IUCN Red List [37]
phylomatic_tree	Get a plant Phylogeny	Phylomatic [33]
plantminer	Search Plantminer	Plantminer [17]
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 [10]
tp_acceptednames	Check for accepted names using Tropicos	Tropicos [16]
tpl_search	Search the Plant List	The Plant List [18]
ubio_namebank	Search uBio	uBio [9]