

## 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", "Abies magnificaa",
  "Rosa californica", "Datura wrighti", "Mimulus bicolour", "Nicotiana glauca",
  "Maddia sativa", "Bartlettia scapposa")
```

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
3	Helanthus annuus	Helianthus annuus	iPlant_TNRS	0.98
1	Pinos contorta	Pinus contorta	iPlant_TNRS	0.96
4	Collomia grandiflorra	Collomia grandiflora	iPlant_TNRS	0.99
5	Abies magnificaa	Abies magnifica	iPlant_TNRS	0.98
10	Rosa californica	Rosa californica	iPlant_TNRS	0.99
9	Datura wrighti	Datura wrightii	iPlant_TNRS	0.98
7	Mimulus bicolour	Mimulus bicolor	iPlant_TNRS	0.98
8	Nicotiana glauca	Nicotiana glauca	iPlant_TNRS	1
6	Maddia sativa	Madia sativa	iPlant_TNRS	0.97
2	Bartlettia scapposa	Bartlettia scaposa	iPlant_TNRS	0.98

```
# 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] "Abies magnifica"       "Rosa californica"      "Datura wrightii"
[7] "Mimulus bicolor"       "Nicotiana glauca"      "Madia sativa"
[10] "Bartlettia scaposa"
```

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 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	Abies magnifica	golden fir
5	Rosa californica	California wildrose
6	Datura wrightii	sacred thorn-apple
7	Mimulus bicolor	yellow and white monkeyflower
8	Nicotiana glauca	tree tobacco
9	Madia sativa	coast tarweed
10	Bartlettia scaposa	Bartlett daisy

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 11 12 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 Infrakingdom
1   Abies magnifica    golden fir Plantae Viridaeplantae Streptophyta
2 Bartlettia scaposa Bartlett daisy Plantae Viridaeplantae Streptophyta
      Division      Subdivision Infradivision      Class Superorder
1 Tracheophyta Spermatophytina Gymnospermae      Pinopsida      <NA>
2 Tracheophyta Spermatophytina Angiospermae Magnoliopsida Asteranae
      Order      Family      Genus
1   Pinales      Pinaceae      Abies
2 Asterales Asteraceae Bartlettia

# 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)

```

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 occurrences
occurr_list <- occurrencelist_many(as.character(allnames$spname), coordinatestatus = TRUE,
  maxresults = 100, removeZeros = TRUE, fixnames = "changealltorig")

# Make a map
p <- gbifmap(occurr_list) + guides(col = guide_legend(title = "", nrow = 3,
  byrow = TRUE)) + theme(legend.position = "bottom", legend.key = element_blank()) +
  coord_equal()
p

```

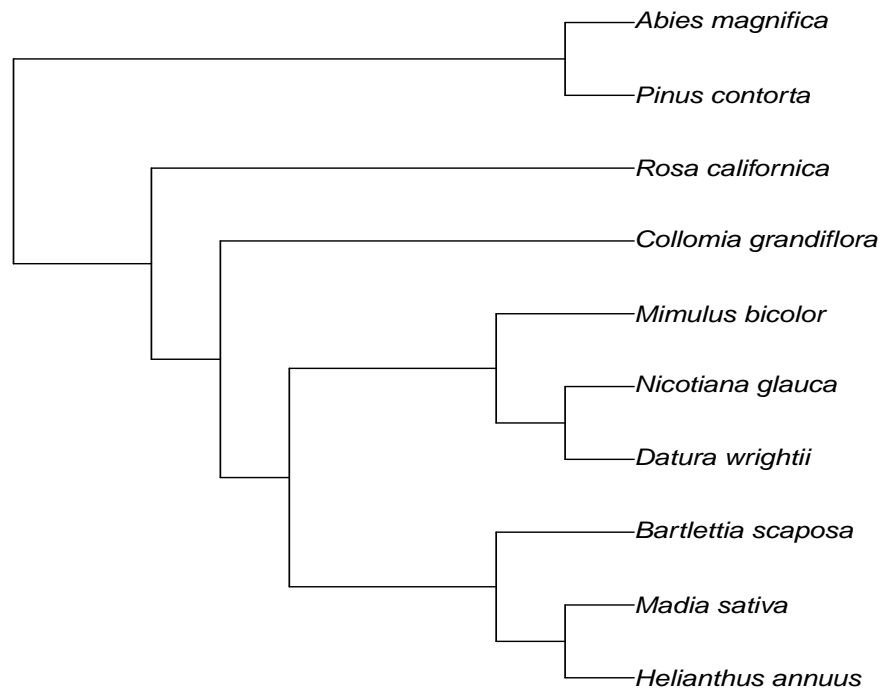


FIG. 1. A phylogeny

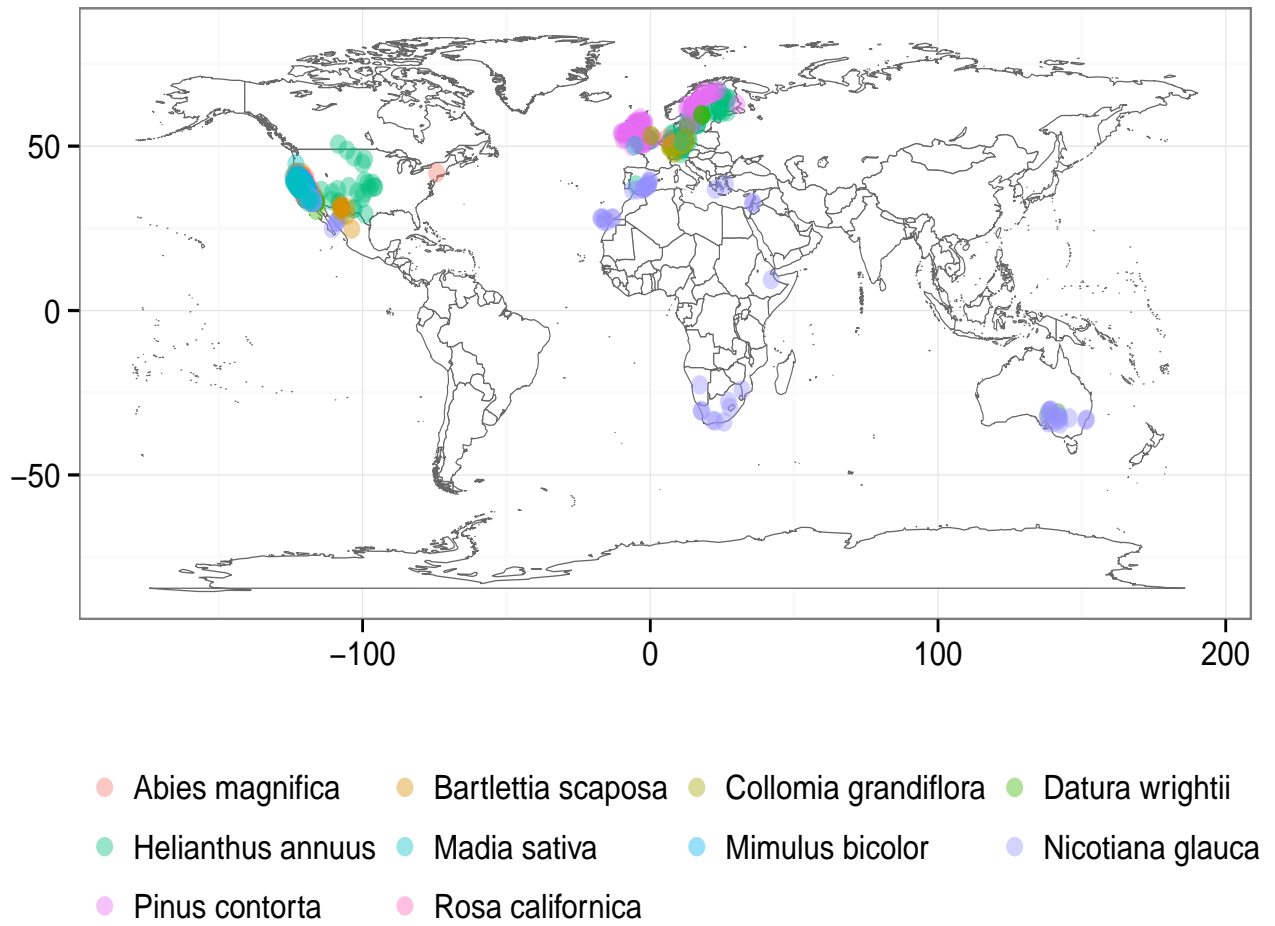


FIG. 2. A map

## 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 [25].

To analyse trait-composition abundance data must be matched with trait databases like [26]. 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 where we have to aggregate trait-data, since our abundance data is on a lower taxonomic level. We are walking the taxonomic ladder for the species in our trait-data upwards and search for matches with our 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](http://www.freshwaterecology.info) database into taxize, which will facilitate trait-based analyses in R.