rphenoscate: Study Case 1

Diego S. Porto, Sergei Tarasov, Caleb Charpentier, and SCATE team

07 February, 2023

In this first study case, we will show how to use *rphenoscape* and *rphenoscate* to perform semantic-aware evolutionary analyses of morphological data. This time, we are going to employ a user-provided data set and an external ontology. We will use the bee data of Porto and Almeida (2021) including information not only from absences/presences of anatomical entities, but also their qualities (e.g., shape, structure, composition, etc.). We will set up appropriate models accounting for trait dependencies based on prior knowledge available from an anatomy ontology, in this case the Hymenoptera Anatomy Ontology (HAO).

STEP 1. Installing and loading the packages.

If you have not installed the package yet, then run the following:

```
remotes::install_github("phenoscape/rphenoscape", build_vignettes = TRUE)
```

You should also install its companion package *rphenoscape* that allows access to the Phenoscape KB.

```
remotes::install_github("uyedaj/rphenoscate", build_vignettes = TRUE)
```

Now, let's load the packages *rphenoscape* and *rphenoscate*.

```
library("rphenoscape")
library("rphenoscate")
```

Let's load some other packages that might be useful as well. If you do not have they installed, please do so. In particular, *ontologyIndex* (Greene et al., 2017) allows us to work with external ontologies provided in .OBO format, the common standard used in the Biological and Biomedical Ontology Foundry. *ontoFAST* (Tarasov et al., 2022) allows us to perform semi-automatic annotations of phylogenetic character statements with anatomy ontology terms.

```
library("ape")
library("phytools")
library("TreeTools")
library("treeplyr")
library("ontologyIndex")
library("ontoFAST")
library("tibble")
library("stringr")
```

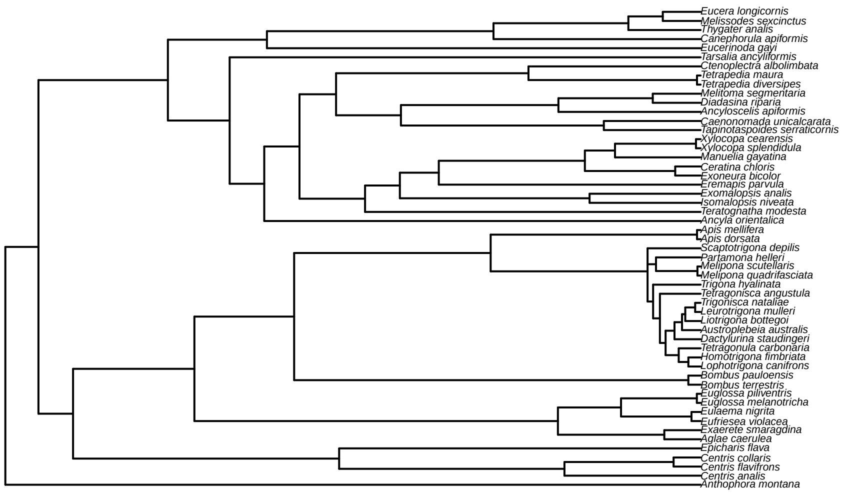
STEP 2. Importing and assembling the data set.

Let's load the data set from Porto and Almeida (2021). It comprises of 289 characters for 53 taxa of apid bees (Hymenoptera: Apoidea: Apidae).

```r
# Import NEXUS file.
nex <- ReadCharacters(file = "data/bees.nex")

# Extract character state labels.
chars <- attributes(nex)$state.labels
chars <- setNames(chars, colnames(nex))

# Organize character matrix.
mat <- bind_cols(taxon = rownames(nex), as_tibble(nex))
mat <- setNames(mat, c("taxon", colnames(nex)))
```

Stochastic character mapping will be performed using the phylogeny available from the same study. Note that the original tree is not a dated phylogeny. For demonstrative purposes only, this tree was forced to ultrametric using the function 'force.ultrametric' from phytools. Therefore, branch lengths are not the result of a formal dating analysis.

```r
# Import tree file.
tree <- read.tree("data/bee_tree.tre")

# Plot tree.
plot.phylo(tree, cex = 0.3)
```



In the current form, this data set is simply a standard phylogenetic character matrix in NEXUS format. However, with functions from *ontoFAST* it is possible to parse through character statements and extract candidate matches to terms from a user-provided anatomy ontology. For that, we need first to import the HAO .OBO file in R using functions from *ontologyIndex*.

```r
onto <- get_OBO("data/HAO.obo", extract_tags = "everything",
                propagate_relationships = c("BFO:0000050", "is_a"))
```

Let's perform the semi-automatic term annotation.

```r
# Pre-organize the ontology.
onto$parsed_synonyms <- syn_extract(onto)

id_characters <- str_c("CHAR:",1:(dim(mat)[2] - 1))
onto$name_characters <- setNames(names(mat)[-1],id_characters)
onto$id_characters <- id_characters

# Run ontoFAST.
auto_annot <- annot_all_chars(onto)
```

```
## [1] "Doing automatic annotation of characters with ontology terms..."
```

```r
# Extract candidate terms.
cand <- setNames(auto_annot, names(mat)[-1])
```

Although this process is semi-automatic, we still need to refine the annotations. For that, we can use a filter function from *rphenoscate*. Based on expert knowledge, we can set two types of filters: strict and general. A strict filter removes all specific terms from a given set. In this case, a series of unnecessary terms referring to general anatomical regions (e.g. 'margin', 'depression') or general entities from the insect anatomy that do not refer to particular anatomical entities (e.g., 'sclerite', 'tergite', 'sternite'). A general filter removes all terms that contain a given word or expression (e.g., 'length', 'distance'). Here we are trying to get only terms referring to anatomical entities (e.g., 'antenna', 'head', 'maxilla', 'wing'), so this is why we need the filtering step. For demonstrative purposes, let's get a sample of 20 character from the original data set.

```r
# Character sample.
char_sample <- c(1,2,10,19,20,26,37,42,50,51,54,63,64,68,80,87,232,233,239,241)

# Sample of candidate terms.
cand_sample <- cand[char_sample]
```

Then, let's set the filters.

```r
# Refine ontoFAST object with filtering terms.
# Strict filter.
s_terms <- c("margin", "edge","ridge", "carina", "spine", "spur", "angle",
             "depression", "sclerite", "tergite", "sternite")

# General filter.
g_terms <- c("length", "distance", "diameter")
```

Finally, let's process the ontoFAST list of candidate terms to obtain the final term annotations. Note that the filter function will still require the input of an expert to select the preferred terms from the filtered set (i.e., this is an interactive step). For simplicity, let's just import the already filtered final set of terms.

```r
# Process ontoFAST object.
# cand_filter <- process.ontofast(cand_sample, onto, s.filter = TRUE, g.filter = TRUE,
#                                 s.terms = s_terms, g.terms = g_terms)

# For demonstrative purposes, let's just load data with the expert based selection of terms.
cand_filter <- readRDS("data/terms.RDS")
```

Then, let's subset and organize the original data set based on our sample of 20 characters.

```r
# Subset character matrix.
mat_sub <- mat[,c(1, char_sample + 1)]

attributes(mat_sub)$row.names <- mat_sub$taxon

# Subset character state information.
state_data <- chars[char_sample]
```

One difference between standard phylogenetic character matrices, such as this one, and synthetic absence/presence matrices, such as the one produced by OntoTrace (see Study Case 2), is that in the former, multiple characters statements can refer to the same anatomical entity, but not necessarily imply dependencies. For example, one character statement may refer to the shape of the 'antenna' and another one to its color. In principle, there is no prior reason to assume ontological dependency between these two transformational character statements. Nonetheless, multiple characters referring to the same anatomical entity, with or without dependencies, can be represented as a single amalgamated character (see Tarasov, 2019, 2022 for in-depth discussion). Therefore, let's amalgamate all characters referring to the same anatomical entities.

```r
# Relabel character statements temporarily.
# make.treedata does not work properly with duplicated columns!
mat_sub <- setNames(mat_sub, c("taxon", str_c("C", 1:(dim(mat_sub)[2] - 1))))

# Build phylogenetic data set #
td <- make.treedata(tree, mat_sub, name_column = "taxon", as.is = TRUE)

# Relabel character statements using the final set of ontology terms.
names(cand_filter) <- NULL
td$dat <- setNames(td$dat, unlist(cand_filter))

attributes(td$dat)$row.names <- td$phy$tip.label

# Get the dependency matrix using rphenoscate.
dep_mat <- auto_dep_matrix(td, tax.col = FALSE)

# Amalgamate dependent characters.
amal_deps <- amalgamate_deps_gen(td, dep_mat, mode = "check",
                                 state.data = state_data, tax.col = FALSE)
```

```
## Using automatic qualitative type of intial vector phi.
## The intial vector phi is phi=c(1,1)
##
## Using automatic qualitative type of intial vector phi.
## The intial vector phi is phi=c(1,1)
##
## Using automatic qualitative type of intial vector phi.
## The intial vector phi is phi=c(1,1,1,1,1)
```

For this, we built a 'dependency matrix' to first amalgamate characters annotated with the same anatomy term. Some annotated characters will be actually independent from each other (e.g., 'shape' and 'color' of a structure) whereas others will be dependent (e.g., 'presence' and 'shape' or 'color' of a structure). These require different types of evolutionary models (i.e., SMM-ind and EDql; see Tarasov, 2022 for more details). Other types of dependencies among different anatomical entities (i.e., parthood-relations) can be present, as will be demonstrated in the Study Case 2.

Let's recode the amalgamated characters.

```
# Recode the amalgamated characters.
td_comb <- recode_traits_gen(td = td, amal.deps = amal_deps, tax.col = FALSE, as.is = TRUE)
```

STEP 3. Check data set for dependencies.

Now we need to check for possible dependencies across the different anatomical entities represented by one or more (i.e., amalgamated) characters from the previous step. Once again, this can be achieved by using a function from *rphenoscate*, but in this case with an external ontology.

```
# Get the dependency matrix using rphenoscape.
dep_mat2 <- dep_matrix(td_comb, onto, tax.col = FALSE)
```

Checking the dependency matrix so we can see that in this case no further dependencies were found (i.e., all lower diagonal values equal 0) so we can proceed to the next step without recoding again the characters.

```
sum(dep_mat2[lower.tri(dep_mat2)])
```

```
## [1] 0
```

STEP 4. Fitting models of trait evolution.

We are going to fit models using *corHMM* with the wrapper function 'amalgamate_fits_corHMM' from *rphenoscate*.

```
corhmm_fits <- amalgamated_fits_corHMM(td_comb, amal_deps)
```

STEP 5. Sampling histories of trait evolution.

Finally, let's use 'amalgamated_simmaps_corHMM' from *rphenoscate* to sample stochastic character maps.

```
stmaps <- amalgamated_simmaps_corHMM(corhmm_fits, nSim = 100)
names(stmaps) <- colnames(td_comb$dat)
```

Plot some samples of character histories from different traits.

```
par(mfrow = c(2,2), mar = c(0.1,0.1,5.0,0.1))

plotSimmap(stmaps[[3]][[15]], ftype = "off")
```

```
## no colors provided. using the following legend:
##         0         1         2
##   "black" "#DF536B" "#61D04F"
```

```
title(main = names(stmaps)[3], font.main = 2, cex.main = 0.75, line = -0.3)

plotSimmap(stmaps[[4]][[15]], ftype = "off")
```

```
## no colors provided. using the following legend:
##         0         1
##   "black" "#DF536B"
```

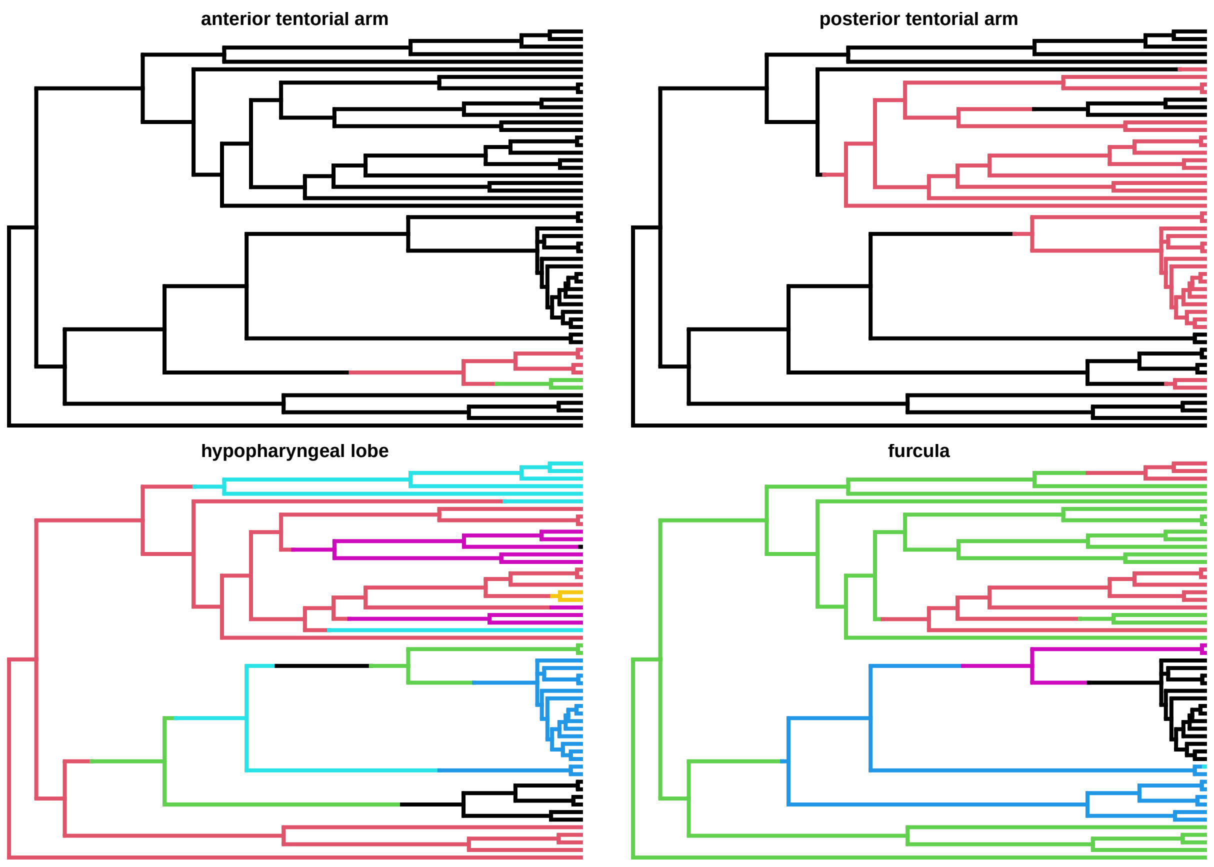
```
title(main = names(stmaps)[4], font.main = 2, cex.main = 0.75, line = -0.3)

plotSimmap(stmaps[[5]][[15]], ftype = "off")
```

```
## no colors provided. using the following legend:
##         0         1         2         3         4         5         6
##   "black" "#DF536B" "#61D04F" "#2297E6" "#28E2E5" "#CD0BBC" "#F5C710"
```

```
title(main = names(stmaps)[5], font.main = 2, cex.main = 0.75, line = -0.3)

plotSimmap(stmaps[[14]][[15]], ftype = "off")
```

```
## no colors provided. using the following legend:
##         0         1         2         3         4         5
##   "black" "#DF536B" "#61D04F" "#2297E6" "#28E2E5" "#CD0BBC"
```

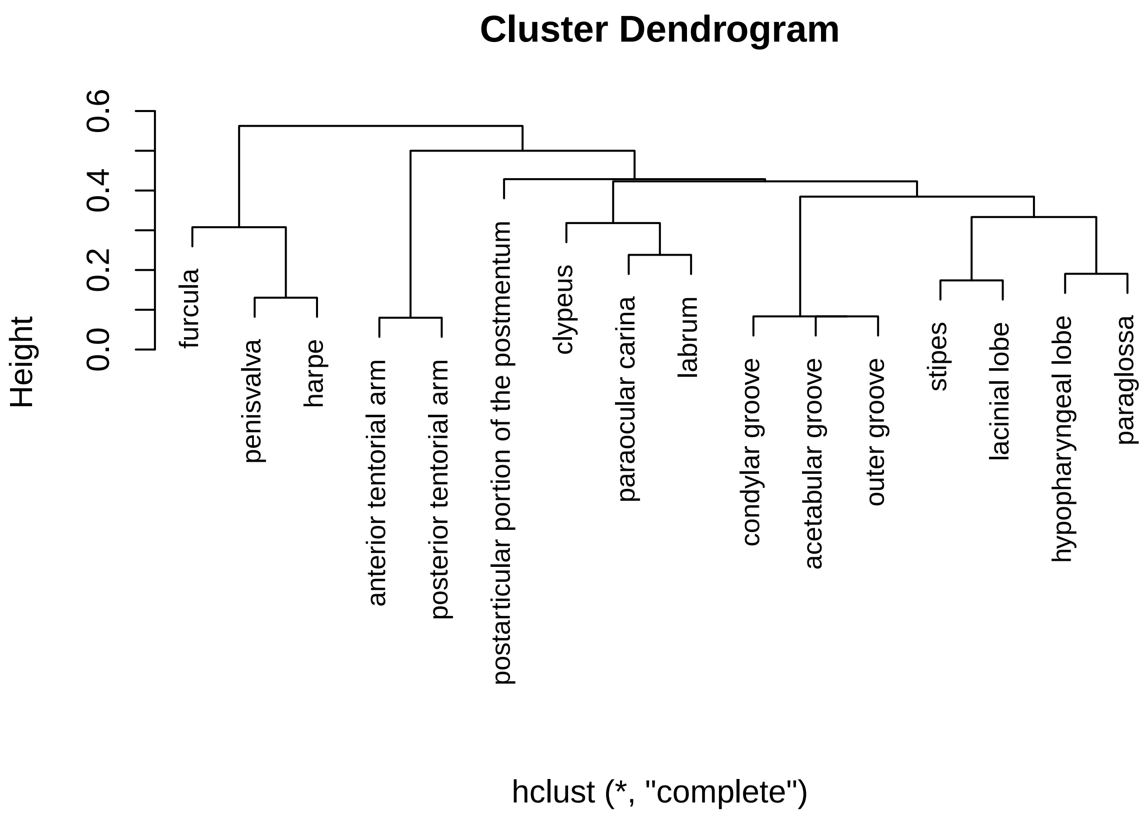
```
title(main = names(stmaps)[14], font.main = 2, cex.main = 0.75, line = -0.3)
```



STEP 6. Exploring semantic properties of the data set.

As the last step, let's build a clustering dendrogram based on the semantic similarity to investigate the relations among ontology terms annotated to the anatomical entities in our data set.

```
plot(makeTraitTree(td_comb, external = TRUE, ONT = onto, method = "cls"), cex = 0.8, xlab = "")
```



References

Greene, D., Richardson, S., and Turro, E. (2017). ontologyx: a suite of r packages for working with ontological data. *Bioinformatics*, 33(7):1104–1106.

Porto, D. S. and Almeida, E. A. (2021). Corbiculate bees (hymenoptera: Apidae): Exploring the limits of morphological data to solve a hard phylogenetic problem. *Insect Systematics and Diversity*, 5(3):2.

Tarasov, S. (2019). Integration of anatomy ontologies and evo-devo using structured markov models suggests a new framework for modeling discrete phenotypic traits. *Systematic biology*, 68(5):698–716.

Tarasov, S. (2022). New phylogenetic markov models for inapplicable morphological characters. *bioRxiv*.

Tarasov, S., Mikó, I., and Yoder, M. J. (2022). ontofast: an r package for interactive and semi-automatic annotation of characters with biological ontologies. *Methods in Ecology and Evolution*, 13(2):324–329.