rphenoscate: Study Case 3

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

07 February, 2023

PART 1. Benchmarking inferred character matrices.

In this third study case, we will be using *rphenoscate* and *rphenoscape* to assemble a phylogenetic character matrix from annotated phenotypes of fishes available in the Phenoscape Knowledgebase (KB). Particularly, we will be using the data set of Dillman et al. (2016) as a benchmark, which consists of 173 species of anostomoid fishes and 463 morphological characters. We will be comparing the 'original' data set (i.e., character matrix retrieved from the original paper) with an 'inferred' data set built from phenotype annotations of the same study. The aim is to demonstrate that 'synthetic' matrices retain most of the phylogenetic information of standard character matrices thus allowing phylogenetic inferences with ontology annotated morphological data.

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, *TreeTools* (Smith, 2019) allows us to import character matrices in NEXUS format and calculate some information metrics from phylogenetic characters. *TreeDist* (Smith, 2020b) allows us to calculate additional information metrics from phylogenetic trees.

```
library("ape")
library("TreeTools")
library("TreeDist")
library("tibble")
library("stringr")
library("ggplot2")
library("viridis")
```

STEP 2. Assembling the data set from a given study.

First, let's retrieve the phylogenetic character matrix from Dillman et al. (2016).

```
# Retrieve the phylogenetic character matrix from Dillman et al (2016).
studies <- get_studies()

# Get a particular study # (Change this part to get a particular study).
study <- studies$id[studies$label == 'Dillman et al. (2016)']

# Get NeXML data.
selected_study <- get_study_data(study)

# Build the original character matrix.
char_mat <- RNeXML::get_characters(selected_study[[1]])
```

Sometimes, data sets might have rows and/or columns represented as IRIs or character IDS instead of actual human-readable labels. Let's check for that.

```
# Get rownames and colnames from data set.
row_mat <- rownames(char_mat)
col_mat <- colnames(char_mat)

# Check rownames and colnames from data set.
row_mat[1:3]
```

```
## [1] "Abramites hypselonotus" "Anostomoides laticeps"  "Anostomus anostomus"
```

```
col_mat[1:3]
```

```
## [1] "Alignment of border between frontal and parietal on dorsal surface of cranium"
## [2] "Alignment of dorsal process of fourth epibranchial"
## [3] "Alignment of ectopterygoid"
```

If rownames and/or colnames are IRIs and/or character IDs, then run the code below to extract the original labels.

```
# Get metadata the original character matrix.
selected_study_meta <- get_char_matrix_meta(selected_study[[1]])

# Get rownames and colnames from data set.
row_mat <- selected_study_meta[[1]]$id_taxa$label[
  match(rownames(char_mat), selected_study_meta[[1]]$id_taxa$otu)]

col_mat <- selected_study_meta[[1]]$id_entities$label[
  match(colnames(char_mat),selected_study_meta[[1]]$id_entities$char)]
```

STEP 3. Getting all semantic phenotypes from a given study.

Now, let's retrieve all phenotype annotations from the Phenoscape KB corresponding to characters statements in the Dillman et al. (2016) study. As mentioned elsewhere, Phenoscape KB contains expert annotated

phenotypes for several studies of fishes. In each study, character statements were converted to semantic statements using a formal syntax, the Entity-Quality syntax (EQ). Character statements can be manually annotated with terms from an anatomy ontology for the anatomical entities (in this case UBERON) and a phenotype ontology for the qualities of entities (in this case PATO) using the software Phenex (Balhoff et al., 2010, 2014).

```
# Retrieve all semantic phenotypes from the Phenoscape KB.
# Get all phenotype data from the study.
phenotypes <- get_phenotypes(study = study)

# Convert data to a phenotype object (Warning: 10~30 min).
selected_study_obj <- as.phenotype(phenotypes, withTaxa = TRUE)

# Get all character information from the phenotype object.
selected_study_chars <- chars(selected_study_obj)

# Filter characters based on target study.
selected_study_chars_filter <- lapply(selected_study_chars, function(x) x[
  grepl(selected_study_chars$study.id, pattern = study)] )
```

The phenotype annotations retrieved from the Phenoscape KB can be found in the 'selected_study_obj'. Each element of this object contains information for a given phenotype annotation, including phenotype ID, label, taxonomic distribution, etc. The object 'selected_study_chars_filter' contains information for character and character state data associated with each phenotype annotation, including character and character state labels as in the reference study (i.e., Dillman et al., 2016).

STEP 4. Obtaining mutual exclusivity information.

Now that we have all phenotype annotations from Dillman et al. (2016), we need to assemble them in a character matrix. The core idea here is that each phenotype annotation (i.e, a semantic statement expressed through the EQ syntax) might correspond to an individual character statement about a quality of a taxon or group of taxa. In simple terms, characters in a standard character matrix can be seen as collections of two or more semantic statements. For example, the semantic statements 'anatomical region and (anterior_to some pelvic fin) circular' and 'anatomical region and (anterior_to some pelvic fin) concave' compose the phylogenetic character 'Form of prepelvic region of ventral body surface'. To cluster phenotype annotations into characters, *rphenoscate* parses information from the Phenoscape KB to classify pairs of phenotypes into exclusivity classes using functions from *rphenoscape*. For more details, refer to the documentation of the 'mutually_exclusive' function of *rphenoscape*. In a nutshell, 'strong_exclusivity' indicates putative pairs of mutually exclusive phenotypes that might correspond to alternative character states of a phylogenetic character. Alternatively, 'strong_compatibility' indicates phenotypes that describe redundant information, therefore, states that can be fused.

Let's obtain the exclusivity classes for all phenotypes annotated in the Dillman et al. (2016).

```
# Determine mutual exclusivity (Warning: 10~30 min).
exclusivity <- mutually_exclusive(phenotypes$id, studies = study)
```

STEP 5. Building characters from phenotype descriptions.

Then, let's use the information on mutual exclusivity classes to extract clusters of semantic statements and build putative characters.

```
# Extract character clusters.
CH <- extract.chars(exclusivity)

# Build characters.
CH_selected_study <- build.chars(CH, selected_study_chars_filter)
```

The object 'CH_selected_study' includes information on putative characters inferred from the exclusivity classes obtained in the previous step. 'solved' comprises phenotype annotations that were clustered as characters as inferred from evidence of 'strong_compatibility' whereas 'unsolved' comprises phenotype annotations that could not be assigned unambiguously to a particular character.

```
# Quick checks.
# Solved characters.
CH_selected_study$solved$chars[1:3]
```

```
## $`Form of attachment of hyohyoidei abductores on urohyal`
## [1] "narrow insertion to anteriormost portion of bone only"
## [2] "longitudinally expanded with broad insertion across entire ventral surface and lateral margins
##
## $`Alignment of border between frontal and parietal on dorsal surface of cranium`
## [1] "border aligned nearly directly transversely or with moderate anteromedial alignment"
## [2] "medial portion of border posteromedially aligned, more so in larger individuals"
##
## $`Contact of contralateral parietals along dorsal midline`
## [1] "distinct contact in adults of various sizes"
## [2] "no contact or very minimal contact anteriorly in large specimens only"
```

```
CH_selected_study$solved$tokens[1:3]
```

```
## $`Form of attachment of hyohyoidei abductores on urohyal`
## [1] "0" "1"
##
## $`Alignment of border between frontal and parietal on dorsal surface of cranium`
## [1] "0" "1"
##
## $`Contact of contralateral parietals along dorsal midline`
## [1] "0" "1"
```

```
# Number of solved characters.
length(CH_selected_study$solved$chars)
```

```
## [1] 421
```

```
# Unsolved characters.
CH_selected_study$unsolved$chars[1:3]
```

```
## $`Form of A3 portion (=pars stegalis) of adductor mandibulae`
## [1] "with extensive origin along lateral surface of mesopterygoid and metapterygoid and broad inserti
##
## $`Form of posterior portion of A2 portion (= pars mallaris) of adductor mandibulae`
```

```
## [1] "extends medial of levator arcus palatini with partial origin on hyomandibula"
##
## $`Form of anterior portion of A2 (= pars mallaris) and A3 (=pars stegalis) portions of adductor mandi
## [1] "muscles dorsally expanded with contact of these muscles with each other dorsal to tendon that a
```

```
CH_selected_study$unsolved$tokens[1:3]
```

```
## $`Form of A3 portion (=pars stegalis) of adductor mandibulae`
## [1] "0"
##
## $`Form of posterior portion of A2 portion (= pars mallaris) of adductor mandibulae`
## [1] "0"
##
## $`Form of anterior portion of A2 (= pars mallaris) and A3 (=pars stegalis) portions of adductor mandi
## [1] "0"
```

```
# Number of unsolved characters.
length(CH_selected_study$unsolved$chars)
```

```
## [1] 30
```

STEP 6. Building character matrices.

Finally, let's build a synthetic character matrix based on the characters inferred from exclusivity information.

```
# Get set of taxa.
tax <- unique(row_mat)

# Build character matrix for inferred characters.
char_mat_infer <- build.matrix(tax, selected_study_obj,
                               selected_study_chars_filter, CH_selected_study)
```

STEP 7. Phylogenetic inferences with the original and synthetic character matrices.

Now, we can finally compare the original and inferred character matrices to assess how well semantic phenotypes carry the phylogenetic properties of morphological data as observed in standard character matrices.

```
# Copy phylogenetic matrices.
m_original <- char_mat
m_inferred <- char_mat_infer
```

Sometimes, data sets might contain some duplicated rows (taxa) due to taxonomic changes. Let's check for that.

```
# Check for duplicated taxa.
any(duplicated(row_mat))
```

```
## [1] FALSE
```

If duplicated rows were found, then taxa must be merged and non-matching character states must be coded as polymorphisms. If that is the case, then run the code below.

```
# Get duplicated taxa.
dup <- row.mat[duplicated(row_mat)]

# Merge duplicated taxa.
for(i in 1:length(dup)){

  x <- grep(row_mat, pattern = dup[i])

  y <- apply(m_original[x,],2, function(x) paste0(unique(x), collapse = " and "),
             simplify = F)

  m_original[x[1],] <- y
  m_original <- m_original[-x[2],]

}
```

Then, let's format taxon labels, character matrices entries, and command blocks to perform Bayesian Inference with MrBayes. First, let's format character matrices.

```
# Prepare data for phylogenetic analyses.
# Reorganize taxon labels (for MrBayes).
tax_lab <- str_replace_all(tax, pattern = "\\?", replacement = "")
tax_lab <- str_replace_all(tax_lab, pattern = "\\(", replacement = "")
tax_lab <- str_replace_all(tax_lab, pattern = "\\)", replacement = "")
tax_lab <- str_replace_all(tax_lab, pattern = " ", replacement = "_")

# Recode polymorphisms (manually) from the original matrix (for MrBayes).
m_original[is.na(m_original)] <- "?"
m_original <- apply(m_original, 2, function(x)
  str_replace_all(x, pattern = "NA", replacement = "?"))
m_original <- apply(m_original, 2, function(x)
  str_replace_all(x, pattern = " and ", replacement = ","))
m_original <- apply(m_original, 2, function(x)
  str_replace_all(x, pattern = "^\\?,", replacement = ""))
m_original <- apply(m_original, 2, function(x)
  str_replace_all(x, pattern = ",\\?$", replacement = ""))
m_original <- apply(m_original, 2, function(x)
  str_replace_all(x, pattern = "^(\\d),", replacement = "(\\1,"))
m_original <- apply(m_original, 2, function(x)
  str_replace_all(x, pattern = ",(\\d)$", replacement = ",\\1)"))
rownames(m_original) <- rownames(m_inferred) <- tax_lab
```

Then, set parameters for the MCMC. These can be changed if desired.

```
# Set MCMC parameters (change if desired).
gens = 5000000
runs = 2
chains = 4
```

```
# Build MrBayes command block.
mb_block <- readLines("data/mb_block.nex")
mb_block <- gsub(mb_block, pattern = "&&ngens&&",
                 replacement = format(gens, scientific = FALSE))
mb_block <- gsub(mb_block, pattern = "&&nruns&&", replacement = runs)
mb_block <- gsub(mb_block, pattern = "&&nchains&&", replacement = chains)
mb_block_original <- gsub(mb_block, pattern = "&&filename&&", replacement = "original")
mb_block_inferred <- gsub(mb_block, pattern = "&&filename&&", replacement = "inferred")

# Create directory to run analyses.
dir.create("mrbayes")

# Export nexus files.
write.nexus.data(as.matrix(m_original), format = "standard",
                 interleaved = FALSE, file = "mrbayes/original.nex")
write.nexus.data(as.matrix(m_inferred), format = "standard",
                 interleaved = FALSE, file = "mrbayes/inferred.nex")

# Write MrBayes command block.
write(mb_block_original, file = "mrbayes/run_original.nex")
write(mb_block_inferred, file = "mrbayes/run_inferred.nex")
```

STEP 8. Bayesian Inferences on MrBayes.

Run the analyses of 'run_original.nex' and 'run_inferred.nex' from within the 'mrbayes' folder.

STEP 9. Evaluating phylogenetic properties of data sets.

Now, let's examine the phylogenetic properties of the data. By that, we mean comparing the phylogenetic information content available for tree inference from the 'original' and 'inferred' character matrices. Systematists often construct character matrices in order to infer the phylogenetic relationships among taxa. In other words,one of the major goals of assembling character matrices is tree inference. Therefore, assessing the phylogenetic information content and tree topologies inferred from these matrices is a natural way to assess how useful 'synthetic' character matrices can be for phylogeneticts in general.

First, let's measure the cladistic information content sensu Steel and Penny (2005) from all characters in the 'original' and 'inferred' character matrices. For that, we will use a function from *TreeTools* (Smith, 2019). For more discussions about information theory used in the context of tree distance metrics see Smith (2020a).

For importing the consensus trees and tree samples from all analyses, run the following chunk of code:

```
# This chunk of code imports the tree samples and consensuses obtained from MrBayes in this study.
# Import consensus results from MrBayes.
tree_ori <- read.nexus(file = "mrbayes/original.con.tre")
tree_inf <- read.nexus(file = "mrbayes/inferred.con.tre")

# Import tree samples #
trees_ori.r1 <- read.nexus(file = "mrbayes/original.run1.t")
trees_ori.r2 <- read.nexus(file = "mrbayes/original.run2.t")
```

```
trees_inf.r1 <- read.nexus(file = "mrbayes/inferred.run1.t")
trees_inf.r2 <- read.nexus(file = "mrbayes/inferred.run2.t")
```

Then, for calculating the phylogenetic information content from both character matrices, run the following:

```
# Copy original character matrix objects.
c_original <- char_mat
c_inferred <- char_mat_infer

# Reorganize tokens and recode polymorphisms and missings.
c_original[is.na(c_original)] <- "?"
c_original <- apply(c_original, 2, function(x)
  str_replace_all(x, pattern = "(.) and (.)", replacement = "?"))
c_inferred <- apply(c_inferred, 2, function(x)
  str_replace_all(x, pattern = "(.),(.),(.),(.)", replacement = "?"))
c_inferred <- apply(c_inferred, 2, function(x)
  str_replace_all(x, pattern = "(.),(.),(.)", replacement = "?"))
c_inferred <- apply(c_inferred, 2, function(x)
  str_replace_all(x, pattern = "(.),(.)", replacement = "?"))
c_inferred <- apply(c_inferred, 2, function(x)
  str_replace_all(x, pattern = "^\\(", replacement = ""))
c_inferred <- apply(c_inferred, 2, function(x)
  str_replace_all(x, pattern = "\\)$", replacement = ""))

# Calculate information metrics per-character across the matrix.
c_original <- apply(c_original, 2, function(x) CharacterInformation(x) )
c_inferred <- apply(c_inferred, 2, function(x) CharacterInformation(x) )

# Summary.
summary(c_original)
```

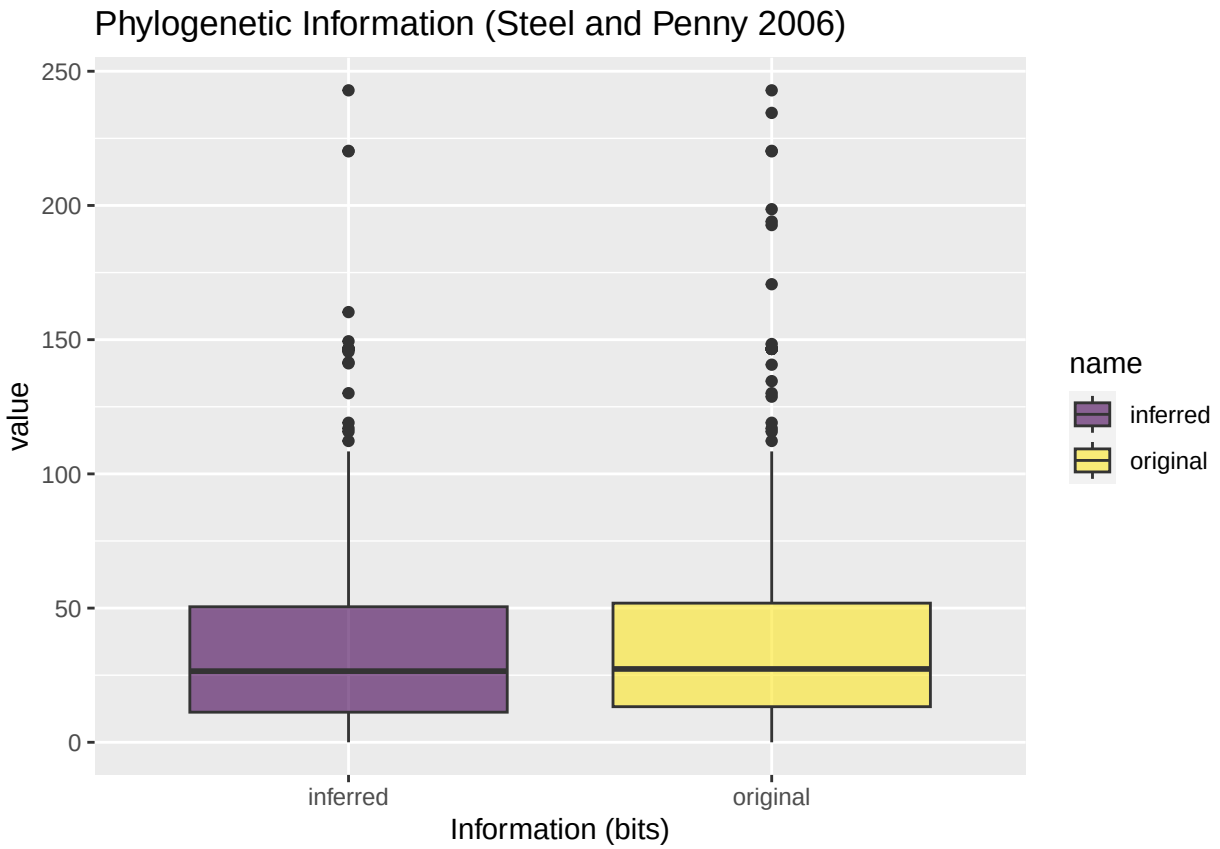
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   13.27   27.32   38.78   51.84  242.89
```

```
summary(c_inferred)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   11.22   26.51   36.57   50.49  242.89
```

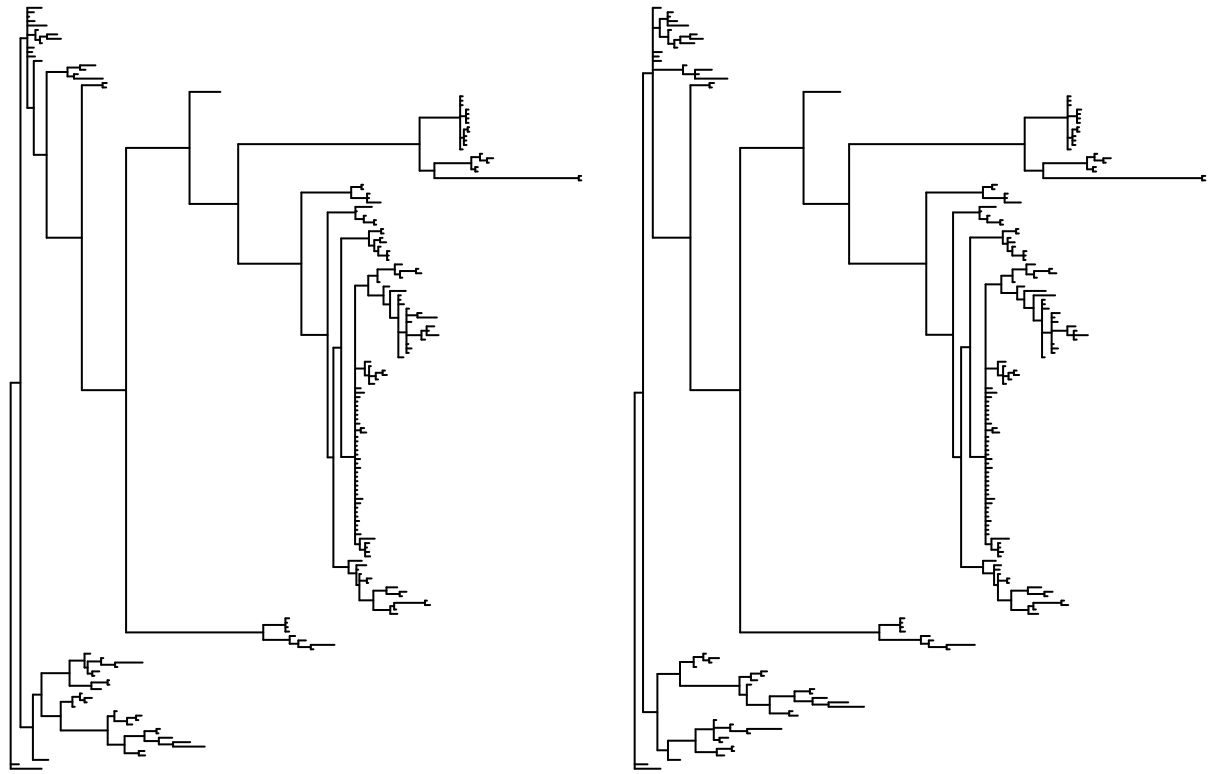
Now, let's make plot comparing the phylogenetic information of the 'original' and 'inferred' matrices.

```
# Build data.frame.
DF1 <- data.frame(value = c(c_original, c_inferred),
                  name = c(rep("original", length(c_original)),
                           rep("inferred", length(c_inferred))))

ggplot(data = DF1, aes(x = name, y = value, fill = name)) + geom_boxplot() +
  scale_fill_viridis(discrete = TRUE, alpha = 0.6) +
  ggtitle("Phylogenetic Information (Steel and Penny 2006)") + xlab("Information (bits)")
```



Now, let's make plot comparing the consensus trees of the 'original' and 'inferred' matrices.

```
par(mfrow = c(1,2), mar = c(0.1,0.1,1.0,0.1))
plot.phylo(tree_ori, show.tip.label = FALSE, edge.width = 1)
plot.phylo(tree_inf, show.tip.label = FALSE, edge.width = 1)
```



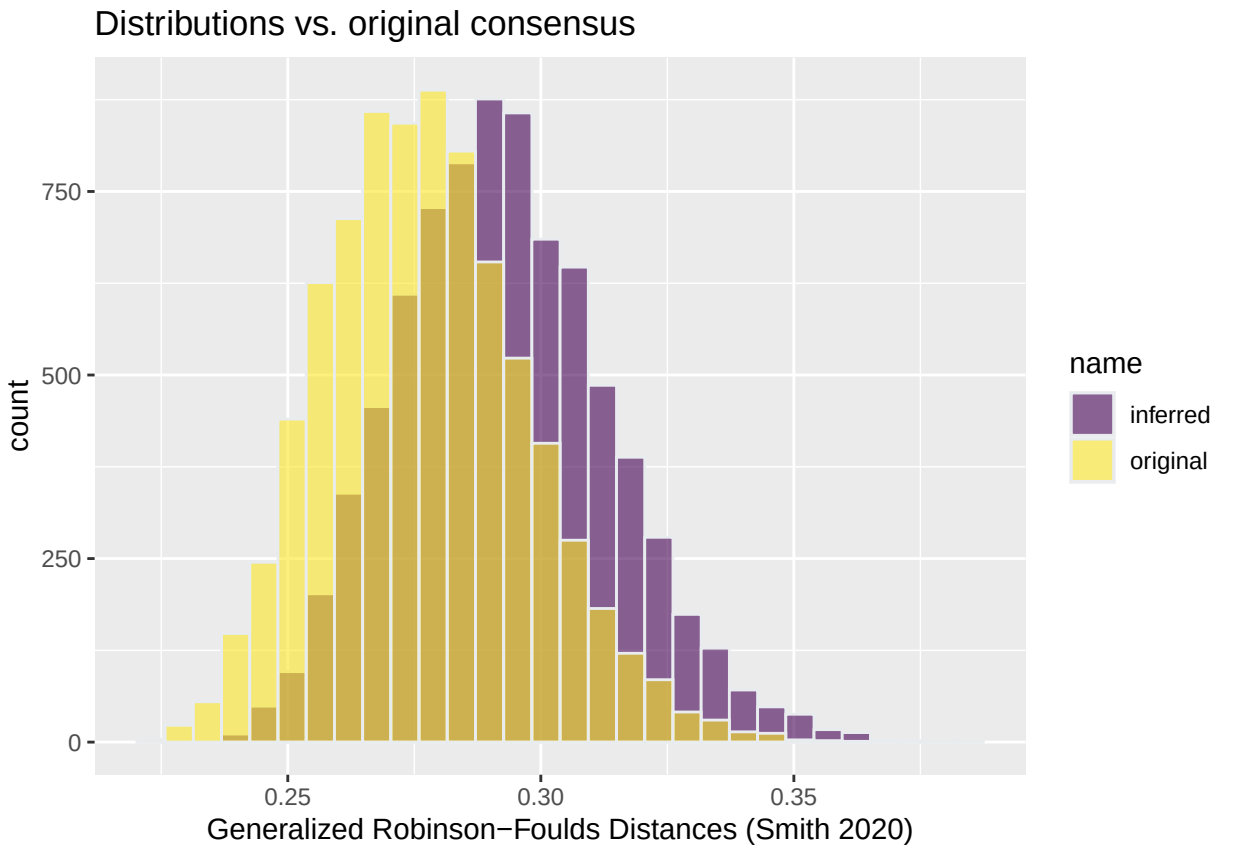
Now, let's measure the phylogenetic information from splits in the consensus trees obtained from both character matrices. For that, we will use functions from *TreeDist* (Smith, 2020b).

```
# Calculate generalize Robinson-Foulds distances among trees.
# Set burnins.
trees_ori.r1 <- trees_ori.r1[-c(1:1001)]
trees_ori.r2 <- trees_ori.r2[-c(1:1001)]
trees_inf.r1 <- trees_inf.r1[-c(1:1001)]
trees_inf.r2 <- trees_inf.r2[-c(1:1001)]

# Organize labels.
names(trees_ori.r1) <- paste0("T", 1:length(trees_ori.r1), ".R1")
names(trees_ori.r2) <- paste0("T", 1:length(trees_ori.r2), ".R2")
names(trees_inf.r1) <- paste0("T", 1:length(trees_inf.r1), ".R1")
names(trees_inf.r2) <- paste0("T", 1:length(trees_inf.r2), ".R2")

# Join chains.
trees_ori_join <- c(trees_ori.r1, trees_ori.r2)
trees_inf_join <- c(trees_inf.r1, trees_inf.r2)

# Calculate generalized RF.
# Non-crossed: each consensus vs. its own distribution.
RF1 <- TreeDistance(tree1 = tree_ori, tree2 = trees_ori_join)
RF2 <- TreeDistance(tree1 = tree_inf, tree2 = trees_inf_join)

# Crossed: each consensus vs. the other distribution.
```

```
RF1_cross <- TreeDistance(tree1 = tree_ori, tree2 = trees_inf_join)
RF2_cross <- TreeDistance(tree1 = tree_inf, tree2 = trees_ori_join)

# Build data.frame.
DF2 <- data.frame(value = c(RF1, RF1_cross), name =
                    c(rep("original", length(RF1)), rep("inferred", length(RF1_cross))))
DF3 <- data.frame(value = c(RF2, RF2_cross), name =
                    c(rep("original", length(RF2)), rep("inferred", length(RF2_cross))))
```

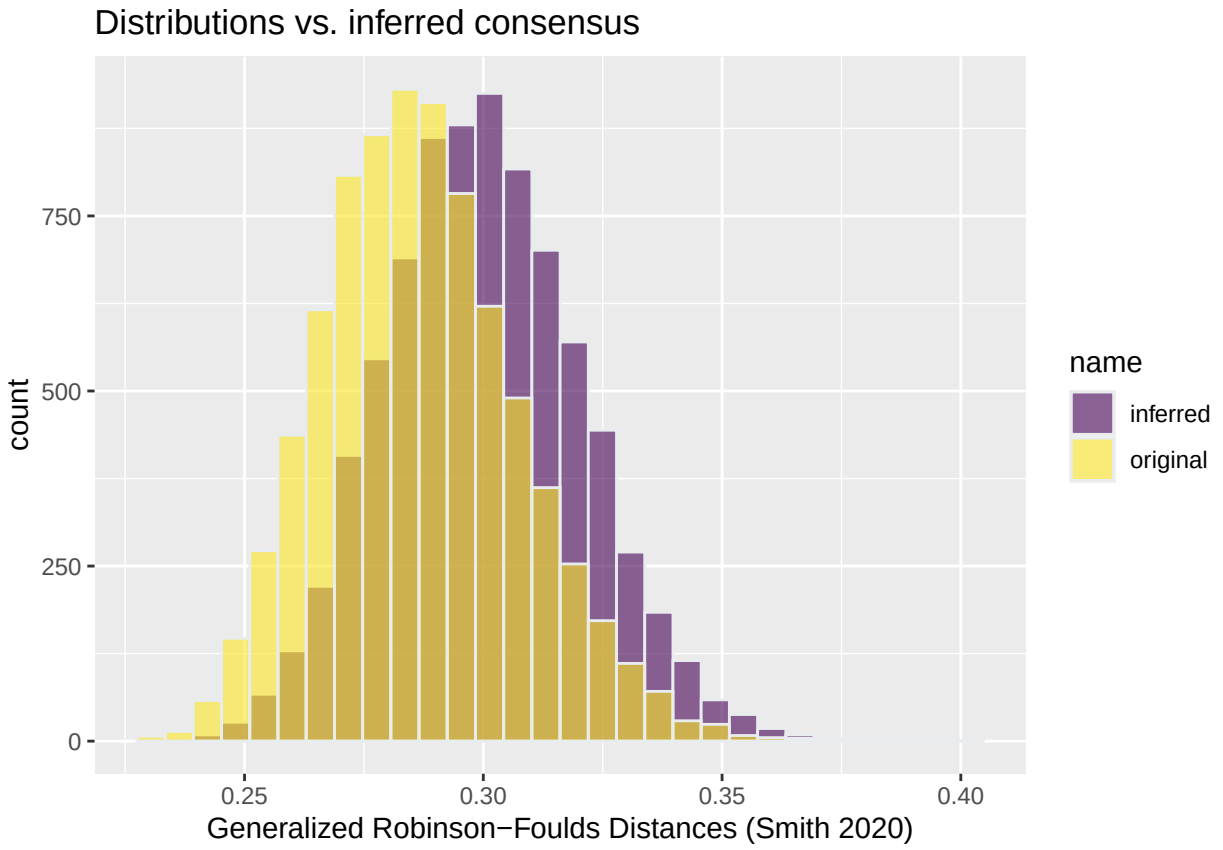
Finally, let's visualize the posterior tree distributions inferred from both data sets. There are several alternatives for visualizing the posterior tree space (for more details see Smith, 2022 for example). Here, let's keep it simple and just use the generalized Robinson-Foulds distances (Smith, 2020a) from each tree topology in the posterior distribution relative to a given consensus tree and plot the histograms. Let's compare each distribution to the consensus tree obtained from the 'original' and 'inferred' data sets.

```
# Consensus from the original matrix.
ggplot(data = DF2, aes(x = value, fill = name)) +
  geom_histogram(color = "#e9ecef", alpha = 0.6, position = 'identity') +
  scale_fill_viridis(discrete = TRUE, alpha = 0.6) +
  ggtitle("Distributions vs. original consensus") +
  xlab("Generalized Robinson-Foulds Distances (Smith 2020)")
```




```
# Consensus from the inferred matrix.
ggplot(data = DF3, aes(x = value, fill = name)) +
  geom_histogram(color = "#e9ecef", alpha = 0.6, position = 'identity') +
```

```
  scale_fill_viridis(discrete = TRUE, alpha = 0.6) +
  ggtitle("Distributions vs. inferred consensus") +
  xlab("Generalized Robinson-Foulds Distances (Smith 2020)")
```



PART 2. Assembling synthetic character matrices.

As a second demonstration of *rphenoscate* and *rphenoscape*, let's retrieve all semantic phenotypes available at the Phenoscape KB for fishes in the family Characidae and assemble a synthetic character matrix.

First, let's get all phenotypes from Characidae.

```
# Get all phenotype data from Characidae.
phenotypes_chara <- get_phenotypes(taxon = "Characidae")

# Convert data to a phenotype object (Warning: >=30 min).
phenotypes_chara_obj <- as.phenotype(phenotypes_chara, withTaxa = TRUE)

# Get all character information from the phenotype object.
phenotypes_chara_char <- chars(phenotypes_chara_obj)
```

Then, let's get the mutual exclusivity information using *rphenoscape*.

```
exclusivity_pk_full <- mutually_exclusive(phenotypes_chara$id)
```

And then assemble a synthetic character matrix from semantic phenotypes.

```
# Extract character clusters.
CH_chara <- extract.chars(exclusivity_pk_full)

# Build characters.
CH_clusters <- build.chars(CH_chara, phenotypes_chara_char)

# Get all species names.
tax_chara <- lapply(phenotypes_chara_obj, function(x) x$taxa$label )
tax_chara <- unique(unlist(tax_chara))

# Filter species names (can take some time!).
#chara <- logical()
#k = 1
#for(i in k:length(tax_chara)){
#
#  chara[i] <- is_descendant(term = "Characidae", candidates = tax_chara[i])
#
#}
#tax_chara <- tax_chara[chara]
#tax_chara <- tax_chara[!is.na(tax_chara)]

# Import filtered taxon names.
tax_chara <- readRDS("data/tax_chara.RDS")

# Building a character matrix.
M <- build.matrix(tax_chara, phenotypes_chara_obj, phenotypes_chara_char, CH_clusters)
```

Finally, let's filter the data set and recode states as binary just to emphasize cells with data vs. non-data, irrespective of the actual character states.

```
# Organize data.
M_org <- M
M_org[M_org != "?"] <- 1
M_org[M_org == "?"] <- 0
M_inf <- as.matrix(type.convert(M_org, as.is = "numeric"))

# Filter taxa and characters with no information.
v1 <- !apply(M_inf, 1, sum) == 0
v2 <- !apply(M_inf, 2, sum) == 0
M_inf <- M_inf[v1,]
M_inf <- M_inf[,v2]

# Check matrix size.
dim(M_inf)
```

```
## [1] 524 739
```

```
# Get taxon coverage.
tax_cover <- round(apply(M_inf, 1, sum)/dim(M_inf)[2],4)
tax_cover[1:5]
```

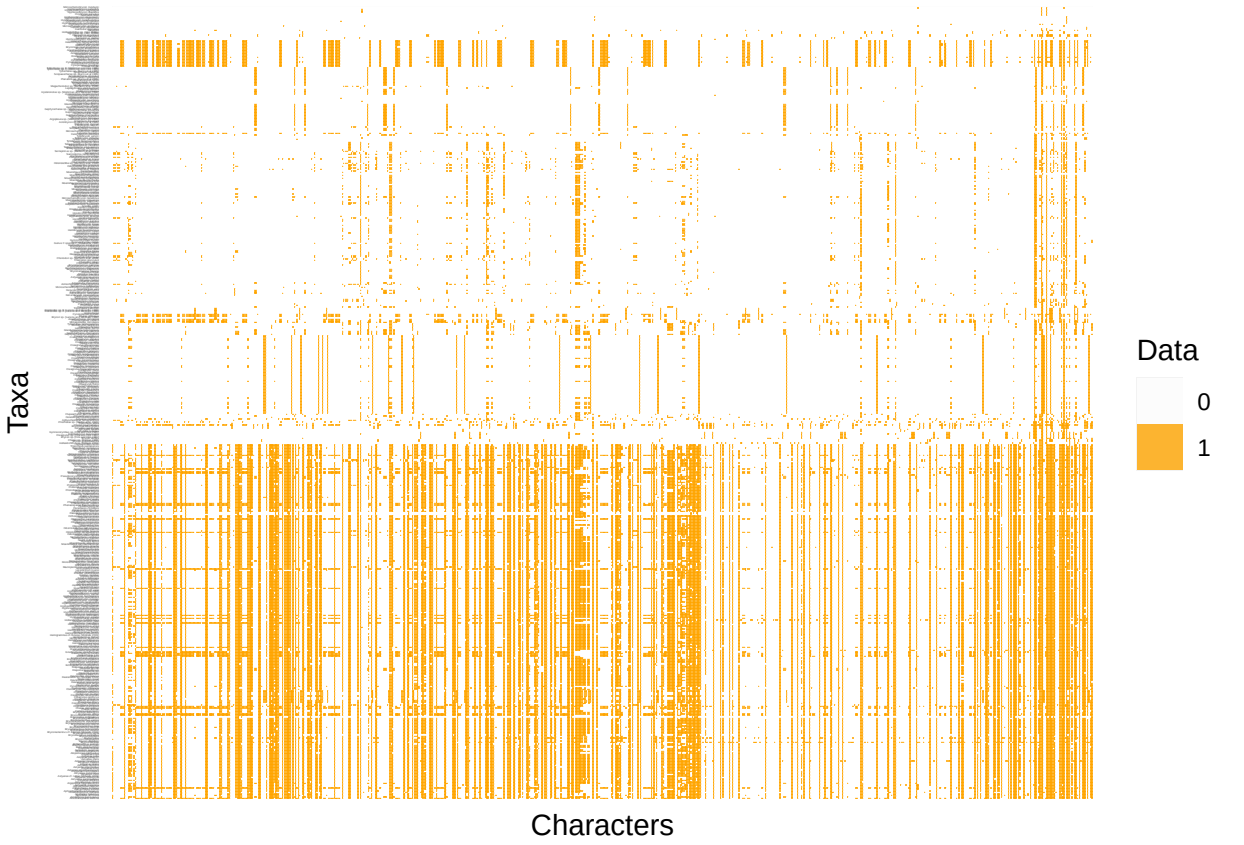
```
## Acestrocephalus sardina  Acrobrycon ipanquianus      Agoniates anchovia
##                  0.5304                  0.3965                  0.3491
##     Agoniates halecinus  Amazonspinther dalmata
##                  0.3356                  0.2991
```

```
# Get character coverage.
char_cover <- round(apply(M_inf, 2, sum)/dim(M_inf)[1],4)
char_cover[1:5]
```

```
##     C1     C2     C3     C4     C5
## 0.3225 0.0401 0.0401 0.0153 0.0401
```

And then plot the synthetic character matrix.

```
# Build a Heatmap and export figure.
# Build data set.
M_plot <- expand.grid(X = rownames(M_inf), Y = colnames(M_inf))
M_plot$Z <- factor(as.vector(M_inf))

ggplot(M_plot, aes(x = Y, y = X, fill = Z)) + geom_tile(alpha = 0.8) +
  scale_fill_manual("Data",values = c("grey100", "orange") ) +
  theme(axis.text.y = element_text(size = 1, angle = 0, hjust = 1),
        axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.ticks.y = element_blank()) +
  ylab("Taxa") + xlab("Characters")
```



References

Balhoff, J. P., Dahdul, W. M., Dececchi, T. A., Lapp, H., Mabee, P. M., and Vision, T. J. (2014). Annotation of phenotypic diversity: decoupling data curation and ontology curation using phenex. *Journal of biomedical semantics*, 5(1):1–5.

Balhoff, J. P., Dahdul, W. M., Kothari, C. R., Lapp, H., Lundberg, J. G., Mabee, P., Midford, P. E., Westerfield, M., and Vision, T. J. (2010). Phenex: ontological annotation of phenotypic diversity. *PLoS One*, 5(5):e10500.

Dillman, C. B., Sidlauskas, B. L., and Vari, R. P. (2016). A morphological supermatrix-based phylogeny for the neotropical fish superfamily anostomoidea (ostariophysi: Characiformes): phylogeny, missing data and homoplasy. *Cladistics*, 32(3):276–296.

Smith, M. R. (2019). *TreeTools: create, modify and analyse phylogenetic trees*. Comprehensive R Archive Network. R package version 1.7.3.

Smith, M. R. (2020a). Information theoretic generalized robinson-foulds metrics for comparing phylogenetic trees. *Bioinformatics*, 36(20):5007–5013.

Smith, M. R. (2020b). *TreeDist: distances between phylogenetic trees. R package version 2.4.1*.

Smith, M. R. (2022). Robust analysis of phylogenetic tree space. *Systematic Biology*, 71(5):1255–1270.

Steel, M. and Penny, D. (2005). Maximum parsimony and the phylogenetic information in multistate characters. *Parsimony, phylogeny and genomics*, pages 163–178.