Lab 3 - Gr. 14 - Bioinformatics (732A93)

Julius Kittler (julki092), Stefano Toffol (steto820), Saewon Jun (saeju204), Maximilian Pfundstein (maxpf364)

Assignment 1

Using the script http://ape-package.ird.fr/APER/APER2/SylviaWarblers.R obtain the Sylvia warblers phylogeny (the script saves in the file `sylvia_nj_k80.tre`). The geographical range data can be found in http://ape-package.ird.fr/APER/APER2/sylvia_data.txt and in the script is referenced as DF$geo.range. Notice that one tip is removed due to missing dat

`tr <- drop.tip(tr, "Chamaea_fasciata")`

and the data has to be ordered by the tips of the phylogeny

`DF <- sylvia.eco[tr$tip.label, ]`

WARNING: Running the script bare might result in errors and very long running times.

Choose only the lines that you actually need!

1.1

Question: Explain all the steps in the script required to obtain the phylogeny and trait data.

The keep it clean we keep track of everything as a bullet point list:

Setup

- We moved all library imports to the top of our RMarkdown file.
- Install ClustalW from http://www.clustal.org/download/current/
- Add ClustalW to the PATH variable of your system
- Install `phyloch` with `install_github("fmichonneau/phyloch")`
- http://www.christophheibl.de/Rpackages.html
- Next error: /usr/local/bin/mafft
- Install 'MAFFT' from https://mafft.cbrc.jp/alignment/software/
- Use the `path`parameter to point to the executable OR
- Add to PATH
- PhyML 3.0 must be installed
- http://www.atgc-montpellier.fr/phyml/download.php
- Point to the executable in the execpath arguments or add to PATH.

Script

```r
# ------------------------------------------------------------------------------
# Question 1.1
# ------------------------------------------------------------------------------

#source("SylviaWarblers.R")
# As we have to explain the code line by line we will not source it but paste it
# here do add comments.
```

```r
###
### Chapter 3
###

# These line create a vector whioch contains 'Z73494' followed by 'AJ5345' with
# ascending numbers at the end ranging from 26 to 49. Then the GenBank database
# is searched and the result saved in 'sylvia.seq' which holds 25 results.
x <- paste("AJ5345", 26:49, sep = "")
x <- c("Z73494", x)
sylvia.seq <- read.GenBank(x)

# 'clustal()' alignes a set of nucletotide sequences. The programm ClustalW must
# be installed locally for this to work.
sylvia.clus <- clustal(sylvia.seq, exec = "/Users/flennic/Downloads/clustalw2")

# MAFFT must be installed. For windows the easiest way is to point to the exe-
# cutable. As it's > 60MB of size, it's not included in the git repository.
# On Linux or macOS this might work out of the box, simply remove the 'path'
# argument.
# MAFFT is used for sequence and profile aligning. It seems like that both
# clustal and mafft return the same result.
sylvia.maff <- mafft(sylvia.seq)
# This function checks if two R objects are equal, which retruns TRUE, so our
# guess was correct.
#identical(sylvia.clus[x, ], sylvia.maff[x, ])

# Add's the attribute 'species' to a new object.
taxa.sylvia <- attr(sylvia.seq, "species")
# The names we mentioned above are assigned to this new object.
names(taxa.sylvia) <- names(sylvia.seq)
# This removes the previous object read from the GenBank.
rm(sylvia.seq)
# The first and the 24th entry get names.
taxa.sylvia[1] <- "Sylvia_atricapilla"
taxa.sylvia[24] <- "Sylvia_abyssinica"

# Now we read from the text file which was provided via a link.
# It holds the geographical range data.
# Note that we changed the name of the file.
sylvia.eco <- read.table("SylviaData.txt")
# Shows the structure of the importet data.frame. It has 26 obervations and 3
# variables.
#str(sylvia.eco)
# Displays the rownames of the data.frame which are a bunch of names 'Sylvia_*'.
#rownames(sylvia.eco)
# We save the three objects to the file called 'sylvia.RData'.
# Note that we added an 'S' to 'sylvia.cluS' as it was misspelled.
save(sylvia.clus, taxa.sylvia, sylvia.eco,
     file = "sylvia.RData")

###
### Chapter 5
###
```

```r
# These functions create a matrix of parwise distances from DNA sequences using
# a model of DNA evolution (taking from Help file).
# The calls also have the argument 'pairwise.deletion' which is set to TRUE.
# This deletes the sites with missing data in a pairwise way.
# The model parameter specifies which model is to be used. As 'K80' is the
# default model, this parameter is not specified for the first call.
# A description of the models can be found in the Help file.
syl.K80 <- dist.dna(sylvia.clus, pairwise.deletion = TRUE)
syl.F84 <- dist.dna(sylvia.clus, model = "F84", p = TRUE)
syl.TN93 <- dist.dna(sylvia.clus, model = "TN93", p = TRUE)
syl.GG95 <- dist.dna(sylvia.clus, model = "GG95", p = TRUE)

# This just plots a distance matrix and is not needed for saving the tree.
#round(cor(cbind(syl.K80, syl.F84, syl.TN93, syl.GG95)), 3)

syl.JC69 <- dist.dna(sylvia.clus, model = "JC69", p = TRUE)
syl.raw <- dist.dna(sylvia.clus, model = "raw", p = TRUE)

# The following code is used for more plotting which we don't need for obtaining
# the tree.
#layout(matrix(1:2, 1))
#plot(syl.JC69, syl.raw)
#abline(b = 1, a = 0) # draw x = y line
#plot(syl.K80, syl.JC69)
#abline(b = 1, a = 0)

#layout(matrix(1:3, 1))
#for (i in 1:3) {
#    s <- logical(3); s[i] <- TRUE
#    x <- sylvia.clus[, s]
#    d <- dist.dna(x, p = TRUE)
#    ts <- dist.dna(x, "Ts", p = TRUE)
#    tv <- dist.dna(x, "Tv", p = TRUE)
#    plot(ts, d, xlab = "Number of Ts or Tv", col = "blue",
#         ylab = "K80 distance", xlim = range(c(ts, tv)),
#         main = paste("Position", i))
#    points(tv, d, col = "red")
#}

#y <- numeric()
#for (i in 1:3) {
#    s <- logical(3); s[i] <- TRUE
#    y <- c(y, dist.dna(sylvia.clus[, s], p = TRUE))
#}
#g <- gl(3, length(y) / 3)

# Plots the histogram
#histogram(~ y | g, breaks = 20)

# The function nj is doing a neighbor-joining tree estimation
nj.sylvia.K80 <- nj(syl.K80)
nj.sylvia.GG95 <- nj(syl.GG95)
```

```r
# dist.topo calculates the topological distance between two trees (12 here).
# It's not needed for saving the tree, so we uncomment it.
#dist.topo(nj.sylvia.K80, nj.sylvia.GG95)

# Just the unix command getting "Chamaea_fasciata"
#grep("Chamaea", taxa.sylvia, value = TRUE)
f <- function(xx) root(nj(dist.dna(xx, p=TRUE)), "AJ534526")
tr <- f(sylvia.clus)
## same than: tr <- root(nj.sylvia.K80, "AJ534526")
# nj.phylo analyse bipartitions found in series of trees (from documntation)
nj.boot.sylvia <- boot.phylo(tr, sylvia.clus, f, 200,
                             rooted = TRUE)
```

```
##
Running bootstraps:       100 / 200
Running bootstraps:       200 / 200
## Calculating bootstrap values... done.
```

```r
nj.boot.codon <- boot.phylo(tr, sylvia.clus, f, 200, 3,
                            rooted = TRUE)
```

```
##
Running bootstraps:       100 / 200
Running bootstraps:       200 / 200
## Calculating bootstrap values... done.
```

```r
nj.est <- tr
nj.est$tip.label <- taxa.sylvia[tr$tip.label]

# The plot is not needed
#plot(nj.est, no.margin = TRUE)
#nodelabels(round(nj.boot.sylvia / 200, 2), bg = "white")
#add.scale.bar(length = 0.01)

# Saves the tree to the file
write.tree(nj.est, "sylvia_nj_k80.tre")

# Writes 25 sequences to a file. Length is 1143.
write.dna(sylvia.clus, "sylvia.txt")

# Calls PhyML and fits 28 models of DNA evolution. They're saved to disk and
# in R returned as a vector. The log-likelihood is saved in this vector.
phyml.sylvia <- phymltest("sylvia.txt", execname = "/Users/flennic/Downloads/PhyML31")

# Again not needed for obtaining the tree
#summary(phyml.sylvia)
#plot(phyml.sylvia, col = "black")

# Read the tree from the txt.
TR <- read.tree("sylvia.txt_phyml_tree.txt")

# Adding some labels and descriptions to the tree
mltr.sylvia <- TR[[28]]
mltr.sylvia$tip.label <- taxa.sylvia[mltr.sylvia$tip.label]
mltr.sylvia <- root(mltr.sylvia, "Chamaea_fasciata")
```

```r
#plot(mltr.sylvia, no.margin = TRUE)
#add.scale.bar(length = 0.01)

# The tip is dropped as explained in the exercise
tr.ml <- drop.tip(mltr.sylvia, "Chamaea_fasciata")
res <- vector("list", 9)

# The for loop takes some time. chronopl estimates the node ages of trees by
# using a semi-parametric method based on penalized likelihood (see docu-
# mentation).
for (L in -4:4)
    res[[L + 5]] <- chronopl(tr.ml, 10^L, 12, 16, CV = TRUE)
```

```
## Doing cross-validation
##
  dropping tip 1 / 24
  dropping tip 2 / 24
  dropping tip 3 / 24
  dropping tip 4 / 24
  dropping tip 5 / 24
  dropping tip 6 / 24
  dropping tip 7 / 24
  dropping tip 8 / 24
  dropping tip 9 / 24
  dropping tip 10 / 24
  dropping tip 11 / 24
  dropping tip 12 / 24
  dropping tip 13 / 24
  dropping tip 14 / 24
  dropping tip 15 / 24
  dropping tip 16 / 24
  dropping tip 17 / 24
  dropping tip 18 / 24
  dropping tip 19 / 24
  dropping tip 20 / 24
  dropping tip 21 / 24
  dropping tip 22 / 24
  dropping tip 23 / 24
  dropping tip 24 / 24
## Doing cross-validation
##
  dropping tip 1 / 24
  dropping tip 2 / 24
  dropping tip 3 / 24
  dropping tip 4 / 24
  dropping tip 5 / 24
  dropping tip 6 / 24
  dropping tip 7 / 24
  dropping tip 8 / 24
  dropping tip 9 / 24
  dropping tip 10 / 24
  dropping tip 11 / 24
  dropping tip 12 / 24
  dropping tip 13 / 24
  dropping tip 14 / 24
  dropping tip 15 / 24
  dropping tip 16 / 24
  dropping tip 17 / 24
  dropping tip 18 / 24
  dropping tip 19 / 24
  dropping tip 20 / 24
  dropping tip 21 / 24
  dropping tip 22 / 24
  dropping tip 23 / 24
  dropping tip 24 / 24
## Doing cross-validation
##
  dropping tip 1 / 24
  dropping tip 2 / 24
  dropping tip 3 / 24
  dropping tip 4 / 24
  dropping tip 5 / 24
  dropping tip 6 / 24
  dropping tip 7 / 24
  dropping tip 8 / 24
  dropping tip 9 / 24
  dropping tip 10 / 24
  dropping tip 11 / 24
  dropping tip 12 / 24
  dropping tip 13 / 24
  dropping tip 14 / 24
  dropping tip 15 / 24
  dropping tip 16 / 24
  dropping tip 17 / 24
  dropping tip 18 / 24
  dropping tip 19 / 24
  dropping tip 20 / 24
  dropping tip 21 / 24
  dropping tip 22 / 24
  dropping tip 23 / 24
  dropping tip 24 / 24
## Doing cross-validation
##
  dropping tip 1 / 24
  dropping tip 2 / 24
  dropping tip 3 / 24
  dropping tip 4 / 24
  dropping tip 5 / 24
  dropping tip 6 / 24
  dropping tip 7 / 24
  dropping tip 8 / 24
  dropping tip 9 / 24
  dropping tip 10 / 24
  dropping tip 11 / 24
  dropping tip 12 / 24
  dropping tip 13 / 24
  dropping tip 14 / 24
  dropping tip 15 / 24
  dropping tip 16 / 24
  dropping tip 17 / 24
  dropping tip 18 / 24
  dropping tip 19 / 24
  dropping tip 20 / 24
  dropping tip 21 / 24
  dropping tip 22 / 24
  dropping tip 23 / 24
  dropping tip 24 / 24
## Doing cross-validation
##
  dropping tip 1 / 24
  dropping tip 2 / 24
  dropping tip 3 / 24
  dropping tip 4 / 24
  dropping tip 5 / 24
  dropping tip 6 / 24
  dropping tip 7 / 24
  dropping tip 8 / 24
  dropping tip 9 / 24
  dropping tip 10 / 24
  dropping tip 11 / 24
  dropping tip 12 / 24
  dropping tip 13 / 24
  dropping tip 14 / 24
  dropping tip 15 / 24
  dropping tip 16 / 24
  dropping tip 17 / 24
  dropping tip 18 / 24
  dropping tip 19 / 24
  dropping tip 20 / 24
  dropping tip 21 / 24
  dropping tip 22 / 24
  dropping tip 23 / 24
  dropping tip 24 / 24
## Doing cross-validation
##
  dropping tip 1 / 24
  dropping tip 2 / 24
  dropping tip 3 / 24
  dropping tip 4 / 24
  dropping tip 5 / 24
  dropping tip 6 / 24
  dropping tip 7 / 24
  dropping tip 8 / 24
  dropping tip 9 / 24
  dropping tip 10 / 24
  dropping tip 11 / 24
  dropping tip 12 / 24
  dropping tip 13 / 24
  dropping tip 14 / 24
  dropping tip 15 / 24
  dropping tip 16 / 24
  dropping tip 17 / 24
  dropping tip 18 / 24
  dropping tip 19 / 24
  dropping tip 20 / 24
  dropping tip 21 / 24
  dropping tip 22 / 24
  dropping tip 23 / 24
  dropping tip 24 / 24
## Doing cross-validation
##
  dropping tip 1 / 24
  dropping tip 2 / 24
  dropping tip 3 / 24
  dropping tip 4 / 24
  dropping tip 5 / 24
  dropping tip 6 / 24
  dropping tip 7 / 24
  dropping tip 8 / 24
  dropping tip 9 / 24
  dropping tip 10 / 24
  dropping tip 11 / 24
  dropping tip 12 / 24
  dropping tip 13 / 24
  dropping tip 14 / 24
  dropping tip 15 / 24
  dropping tip 16 / 24
  dropping tip 17 / 24
  dropping tip 18 / 24
  dropping tip 19 / 24
  dropping tip 20 / 24
  dropping tip 21 / 24
  dropping tip 22 / 24
  dropping tip 23 / 24
  dropping tip 24 / 24
## Doing cross-validation
##
  dropping tip 1 / 24
  dropping tip 2 / 24
  dropping tip 3 / 24
  dropping tip 4 / 24
  dropping tip 5 / 24
  dropping tip 6 / 24
  dropping tip 7 / 24
  dropping tip 8 / 24
  dropping tip 9 / 24
  dropping tip 10 / 24
  dropping tip 11 / 24
  dropping tip 12 / 24
  dropping tip 13 / 24
  dropping tip 14 / 24
  dropping tip 15 / 24
  dropping tip 16 / 24
  dropping tip 17 / 24
  dropping tip 18 / 24
  dropping tip 19 / 24
  dropping tip 20 / 24
  dropping tip 21 / 24
  dropping tip 22 / 24
  dropping tip 23 / 24
  dropping tip 24 / 24
## Doing cross-validation
##
  dropping tip 1 / 24
  dropping tip 2 / 24
  dropping tip 3 / 24
  dropping tip 4 / 24
  dropping tip 5 / 24
  dropping tip 6 / 24
  dropping tip 7 / 24
  dropping tip 8 / 24
  dropping tip 9 / 24
  dropping tip 10 / 24
  dropping tip 11 / 24
  dropping tip 12 / 24
  dropping tip 13 / 24
  dropping tip 14 / 24
  dropping tip 15 / 24
  dropping tip 16 / 24
  dropping tip 17 / 24
  dropping tip 18 / 24
  dropping tip 19 / 24
  dropping tip 20 / 24
  dropping tip 21 / 24
  dropping tip 22 / 24
  dropping tip 23 / 24
  dropping tip 24 / 24
```

```r
Lambda <- 10^(-4:4)
CV <- sapply(res, function(x) sum(attr(x, "D2")))
#plot(Lambda, CV / 1e5, log = "x")

# Add the attribute "rates" to the tree with 24 tips and 23 internal nodes
sylvia.chrono <- res[[2]]
rts <- attr(sylvia.chrono, "rates")
#summary(rts)

# Not needed for obtaining the tree
#par(mar = c(2, 0, 0, 0))
#plot(sylvia.chrono, edge.width = 100*rts, label.offset = .15)
#axisPhylo()

# Finally writes the tree to the file.
write.tree(sylvia.chrono, "sylvia.chrono.tre")

###
### Chapter 6
###

### Do we need all of this source code? Is is about getting the traits?!
```

```r
load("sylvia.RData")
nj.est <- read.tree("sylvia_nj_k80.tre")
nj.est <- drop.tip(nj.est, "Chamaea_fasciata")
DF <- sylvia.eco[nj.est$tip.label, ]
table(DF$geo.range, DF$mig.behav)
```

```
##
##             long resid short
##   temp         0     4     0
##   temptrop     9     0     2
##   trop         0     6     0
```

```r
syl.er <- ace(DF$geo.range, nj.est, type = "d")
syl.sym <- ace(DF$geo.range, nj.est, type="d", model="SYM")
anova(syl.er, syl.sym)
```

```
## Likelihood Ratio Test Table
##   Log lik. Df Df change Resid. Dev Pr(>|Chi|)
## 1  -19.945  1
## 2  -18.401  3         2      3.087     0.2136
```

```r
mod <- matrix(0, 3, 3)
mod[2, 1] <- mod[1, 2] <- 1
mod[2, 3] <- mod[3, 2] <- 2
syl.mod <- ace(DF$geo.range, nj.est, type="d", model=mod)

sapply(list(syl.er, syl.sym, syl.mod), AIC)
```

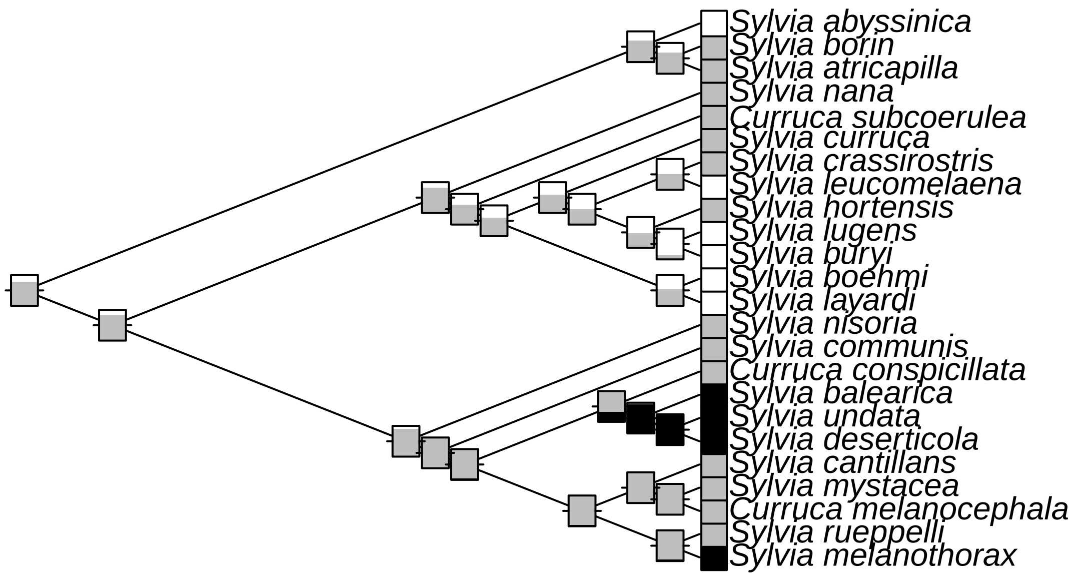
```
## [1] 41.88955 42.80257 40.80257
```

```r
#Q <- syl.mod$index.matrix
#diag(Q) <- 0
#Q[1, 2] <- Q[2, 1] <- syl.mod$rates[1]
#Q[2, 3] <- Q[3, 2] <- syl.mod$rates[2]

#Q[] <- c(0, syl.mod$rates)[Q + 1]
#diag(Q) <- -rowSums(Q)

#P <- matexpo(0.05 * Q)
#rownames(P) <- c("temp", "temptrop", "trop")
#colnames(P) <- rownames(P)

# That's the plot which is nice so we should leave it :)
co <- rep("grey", 24)
co[DF$geo.range == "temp"] <- "black"
co[DF$geo.range == "trop"] <- "white"
plot(nj.est, "c", FALSE, no.margin = TRUE, label.offset = 1)
tiplabels(pch = 22, bg = co, cex = 2, adj = 1)
nodelabels(thermo = syl.mod$lik.anc, cex = 0.8,
           piecol = c("black", "grey", "white"))
```



```r
# This works but do we need this?
#sylvia.chrono <- read.tree("sylvia.chrono.tre")
#yule(sylvia.chrono)
#birthdeath(sylvia.chrono)
#1 - pchisq(2*(-1.034112 - -1.113822), 1)

#x <- sylvia.eco[sylvia.chrono$tip.label, "geo.range"]
#ANC <- ace(x, sylvia.chrono, type = "d", model = mod)
#ANC$lik.anc[1:3, ]
#anc <- apply(ANC$lik.anc, 1, which.max)
#X <- factor(c(x, anc))
# This breaks as we have NAs!
#yule.cov(sylvia.chrono, ~ X)
#1 / (1 + exp(-(-0.0535529)))
#1 / (1 + exp(-(-0.0535529 -1.4608019)))
#1 / (1 + exp(-(-0.0535529 -0.9775966)))

#fsamp <- function(x) sample(length(x), size = 1, prob = x)
#nrep <- 1e3
#Pvls <- numeric(nrep)
#for (i in 1:nrep) {
#    anc <- apply(ANC$lik.anc, 1, fsamp)
#    X <- factor(c(x, anc))
#    Pvls[i] <- yule.cov(sylvia.chrono, ~ X)$Pval
#}
#hist(Pvls, freq = FALSE, main = "")
#lines(density(Pvls))
```

1.2

Question: Analyze the discrete (`type=discrete`) geographical range variable (`DF$geo.range`) using `ape::ace`. Consider different models (parameter `model`). Report on the results and interpret the estimated rates and their standard errors..

Assignment 2

Install the `ade4` R package. Included with it you will find the carnivores dataset, `data(carni70)`.

2.1

Question: Explore the data set and report what can be found in it. Provide some plots.

2.2

Question:

- Analyze the two traits (size and range) with a number of different phylogenetic comparative models.

1. Both traits evolve as independent Brownian motions.
2. The traits evolve as a correlated Brownian motion.
3. Both traits evolve as independent Ornstein{Uhlenbeck processes.
4. The traits evolve as a bivariate Ornstein{Uhlenbeck process (use `mvMORPH` or `mvSLOUCH` but be careful and check under what assumptions the estimation is done).
5. `size` evolves as a Brownian motion and `range` as an Ornstein-Uhlenbeck process adapting to it (use `slouch` or `mvSLOUCH` and be careful about column order).

- Compare the five models and say which one seems to fit better to the data.
- Summarize (in your own words, NOT just provide the printout, as it can be very long) the output under each model. If for some reason the running time will be prohibitive (more than 12 hours) report how long you ran the code and skip the model. Report on any problems you experienced in your study.

WARNING: It is possible to have very long running times for correlated OUOU and OUBM models (the last two), especially on slow computers. Give yourself plenty of reserve time. If you do not manage to complete the last two models in time for submission, submit without them, reporting (with e.g. screenshots) how much time the estimation procedures were running for. Then, you will complete this part on a later mutually agreed on date

Appendix

```r
knitr::opts_chunk$set(fig.width = 7, fig.height = 3, echo = FALSE,
                      warning = FALSE, message = FALSE)

library(ape)
library(knitr)
library(lattice)
# The easiest way to get this package is via
# install_github("fmichonneau/phyloch") more information can be found at
# http://www.christophheibl.de/Rpackages.html
library(phyloch)

# Use this if BiocManager is not installed
#if (!requireNamespace("BiocManager", quietly = TRUE))
#    install.packages("BiocManager")
#library("BiocManager")

# BiocManager packages
```

```r
##############################################################################
# IMPORTANT
# Install ClustalW from http://www.clustal.org/download/current/
# Don't forget to add it to your PATH!
# Install MAFFT version 7 from https://mafft.cbrc.jp/alignment/software/
##############################################################################
# ----------------------------------------------------------------------------
# Question 1.1
# ----------------------------------------------------------------------------

#source("SylviaWarblers.R")
# As we have to explain the code line by line we will not source it but paste it
# here do add comments.

###
### Chapter 3
###

# These line create a vector whioch contains 'Z73494' followed by 'AJ5345' with
# ascending numbers at the end ranging from 26 to 49. Then the GenBank database
# is searched and the result saved in 'sylvia.seq' which holds 25 results.
x <- paste("AJ5345", 26:49, sep = "")
x <- c("Z73494", x)
sylvia.seq <- read.GenBank(x)

# 'clustal()' alignes a set of nucletotide sequences. The programm ClustalW must
# be installed locally for this to work.
sylvia.clus <- clustal(sylvia.seq, exec = "/Users/flennic/Downloads/clustalw2")

# MAFFT must be installed. For windows the easiest way is to point to the exe-
# cutable. As it's > 60MB of size, it's not included in the git repository.
# On Linux or macOS this might work out of the box, simply remove the 'path'
# argument.
# MAFFT is used for sequence and profile aligning. It seems like that both
# clustal and mafft return the same result.
sylvia.maff <- mafft(sylvia.seq)
# This function checks if two R objects are equal, which retruns TRUE, so our
# guess was correct.
#identical(sylvia.clus[x, ], sylvia.maff[x, ])

# Add's the attribute 'species' to a new object.
taxa.sylvia <- attr(sylvia.seq, "species")
# The names we mentioned above are assigned to this new object.
names(taxa.sylvia) <- names(sylvia.seq)
# This removes the previous object read from the GenBank.
rm(sylvia.seq)
# The first and the 24th entry get names.
taxa.sylvia[1] <- "Sylvia_atricapilla"
taxa.sylvia[24] <- "Sylvia_abyssinica"

# Now we read from the text file which was provided via a link.
# It holds the geographical range data.
# Note that we changed the name of the file.
```

```r
sylvia.eco <- read.table("SylviaData.txt")
# Shows the structure of the importet data.frame. It has 26 obvervations and 3
# variables.
#str(sylvia.eco)
# Displays the rownames of the data.frame which are a bunch of names 'Sylvia_*'.
#rownames(sylvia.eco)
# We save the three objects to the file called 'sylvia.RData'.
# Note that we added an 'S' to 'sylvia.cluS' as it was misspelled.
save(sylvia.clus, taxa.sylvia, sylvia.eco,
     file = "sylvia.RData")

###
### Chapter 5
###

# These functions create a matrix of parwise distances from DNA sequences using
# a model of DNA evolution (taking from Help file).
# The calls also have the argument 'pairwise.deletion' which is set to TRUE.
# This deletes the sites with missing data in a pairwise way.
# The model parameter specifies which model is to be used. As 'K80' is the
# default model, this parameter is not specified for the first call.
# A description of the models can be found in the Help file.
syl.K80 <- dist.dna(sylvia.clus, pairwise.deletion = TRUE)
syl.F84 <- dist.dna(sylvia.clus, model = "F84", p = TRUE)
syl.TN93 <- dist.dna(sylvia.clus, model = "TN93", p = TRUE)
syl.GG95 <- dist.dna(sylvia.clus, model = "GG95", p = TRUE)

# This just plots a distance matrix and is not needed for saving the tree.
#round(cor(cbind(syl.K80, syl.F84, syl.TN93, syl.GG95)), 3)

syl.JC69 <- dist.dna(sylvia.clus, model = "JC69", p = TRUE)
syl.raw <- dist.dna(sylvia.clus, model = "raw", p = TRUE)

# The following code is used for more plotting which we don't need for obtaining
# the tree.
#layout(matrix(1:2, 1))
#plot(syl.JC69, syl.raw)
#abline(b = 1, a = 0) # draw x = y line
#plot(syl.K80, syl.JC69)
#abline(b = 1, a = 0)

#layout(matrix(1:3, 1))
#for (i in 1:3) {
#    s <- logical(3); s[i] <- TRUE
#    x <- sylvia.clus[, s]
#    d <- dist.dna(x, p = TRUE)
#    ts <- dist.dna(x, "Ts", p = TRUE)
#    tv <- dist.dna(x, "Tv", p = TRUE)
#    plot(ts, d, xlab = "Number of Ts or Tv", col = "blue",
#         ylab = "K80 distance", xlim = range(c(ts, tv)),
#         main = paste("Position", i))
#    points(tv, d, col = "red")
#}
```

```r
#y <- numeric()
#for (i in 1:3) {
#    s <- logical(3); s[i] <- TRUE
#    y <- c(y, dist.dna(sylvia.clus[, s], p = TRUE))
#}
#g <- gl(3, length(y) / 3)

# Plots the histogram
#histogram(~ y | g, breaks = 20)

# The function nj is doing a neighbor-joining tree estimation
nj.sylvia.K80 <- nj(syl.K80)
nj.sylvia.GG95 <- nj(syl.GG95)

# dist.topo calculates the topological distance between two trees (12 here).
# It's not needed for saving the tree, so we uncomment it.
#dist.topo(nj.sylvia.K80, nj.sylvia.GG95)

# Just the unix command getting "Chamaea_fasciata"
#grep("Chamaea", taxa.sylvia, value = TRUE)
f <- function(xx) root(nj(dist.dna(xx, p=TRUE)), "AJ534526")
tr <- f(sylvia.clus)
## same than: tr <- root(nj.sylvia.K80, "AJ534526")
# nj.phylo analyse bipartitions found in series of trees (from documntation)
nj.boot.sylvia <- boot.phylo(tr, sylvia.clus, f, 200,
                             rooted = TRUE)
nj.boot.codon <- boot.phylo(tr, sylvia.clus, f, 200, 3,
                            rooted = TRUE)
nj.est <- tr
nj.est$tip.label <- taxa.sylvia[tr$tip.label]

# The plot is not needed
#plot(nj.est, no.margin = TRUE)
#nodelabels(round(nj.boot.sylvia / 200, 2), bg = "white")
#add.scale.bar(length = 0.01)

# Saves the tree to the file
write.tree(nj.est, "sylvia_nj_k80.tre")

# Writes 25 sequences to a file. Length is 1143.
write.dna(sylvia.clus, "sylvia.txt")

# Calls PhyML and fits 28 models of DNA evolution. They're saved to disk and
# in R returned as a vector. The log-likelihood is saved in this vector.
phyml.sylvia <- phymltest("sylvia.txt", execname = "/Users/flennic/Downloads/PhyML31")

# Again not needed for obtaining the tree
#summary(phyml.sylvia)
#plot(phyml.sylvia, col = "black")

# Read the tree from the txt.
TR <- read.tree("sylvia.txt_phyml_tree.txt")
```

```
# Adding some labels and descriptions to the tree
mltr.sylvia <- TR[[28]]
mltr.sylvia$tip.label <- taxa.sylvia[mltr.sylvia$tip.label]
mltr.sylvia <- root(mltr.sylvia, "Chamaea_fasciata")
#plot(mltr.sylvia, no.margin = TRUE)
#add.scale.bar(length = 0.01)

# The tip is dropped as explained in the exercise
tr.ml <- drop.tip(mltr.sylvia, "Chamaea_fasciata")
res <- vector("list", 9)

# The for loop takes some time. chronopl estimates the node ages of trees by
# using a semi-parametric method based on penalized likelihood (see docu-
# mentation).
for (L in -4:4)
    res[[L + 5]] <- chronopl(tr.ml, 10^L, 12, 16, CV = TRUE)
Lambda <- 10^(-4:4)
CV <- sapply(res, function(x) sum(attr(x, "D2")))
#plot(Lambda, CV / 1e5, log = "x")

# Add the attribute "rates" to the tree with 24 tips and 23 internal nodes
sylvia.chrono <- res[[2]]
rts <- attr(sylvia.chrono, "rates")
#summary(rts)

# Not needed for obtaining the tree
#par(mar = c(2, 0, 0, 0))
#plot(sylvia.chrono, edge.width = 100*rts, label.offset = .15)
#axisPhylo()

# Finally writes the tree to the file.
write.tree(sylvia.chrono, "sylvia.chrono.tre")

###
### Chapter 6
###

### Do we need all of this source code? Is is about getting the traits?!

load("sylvia.RData")
nj.est <- read.tree("sylvia_nj_k80.tre")
nj.est <- drop.tip(nj.est, "Chamaea_fasciata")
DF <- sylvia.eco[nj.est$tip.label, ]
table(DF$geo.range, DF$mig.behav)

syl.er <- ace(DF$geo.range, nj.est, type = "d")
syl.sym <- ace(DF$geo.range, nj.est, type="d", model="SYM")
anova(syl.er, syl.sym)

mod <- matrix(0, 3, 3)
mod[2, 1] <- mod[1, 2] <- 1
mod[2, 3] <- mod[3, 2] <- 2
syl.mod <- ace(DF$geo.range, nj.est, type="d", model=mod)
```

```
sapply(list(syl.er, syl.sym, syl.mod), AIC)

#Q <- syl.mod$index.matrix
#diag(Q) <- 0
#Q[1, 2] <- Q[2, 1] <- syl.mod$rates[1]
#Q[2, 3] <- Q[3, 2] <- syl.mod$rates[2]

#Q[] <- c(0, syl.mod$rates)[Q + 1]
#diag(Q) <- -rowSums(Q)

#P <- matexpo(0.05 * Q)
#rownames(P) <- c("temp", "temptrop", "trop")
#colnames(P) <- rownames(P)

# That's the plot which is nice so we should leave it :)
co <- rep("grey", 24)
co[DF$geo.range == "temp"] <- "black"
co[DF$geo.range == "trop"] <- "white"
plot(nj.est, "c", FALSE, no.margin = TRUE, label.offset = 1)
tiplabels(pch = 22, bg = co, cex = 2, adj = 1)
nodelabels(thermo = syl.mod$lik.anc, cex = 0.8,
           piecol = c("black", "grey", "white"))

# This works but do we need this?
#sylvia.chrono <- read.tree("sylvia.chrono.tre")
#yule(sylvia.chrono)
#birthdeath(sylvia.chrono)
#1 - pchisq(2*(-1.034112 - -1.113822), 1)

#x <- sylvia.eco[sylvia.chrono$tip.label, "geo.range"]
#ANC <- ace(x, sylvia.chrono, type = "d", model = mod)
#ANC$lik.anc[1:3, ]
#anc <- apply(ANC$lik.anc, 1, which.max)
#X <- factor(c(x, anc))
# This breaks as we have NAs!
#yule.cov(sylvia.chrono, ~ X)
#1 / (1 + exp(-(-0.0535529)))
#1 / (1 + exp(-(-0.0535529 -1.4608019)))
#1 / (1 + exp(-(-0.0535529 -0.9775966)))

#fsamp <- function(x) sample(length(x), size = 1, prob = x)
#nrep <- 1e3
#Pvls <- numeric(nrep)
#for (i in 1:nrep) {
#    anc <- apply(ANC$lik.anc, 1, fsamp)
#    X <- factor(c(x, anc))
#    Pvls[i] <- yule.cov(sylvia.chrono, ~ X)$Pval
#}
#hist(Pvls, freq = FALSE, main = "")
#lines(density(Pvls))

# -----------------------------------------------------------------------------
# Question 1.2
# ------------------------------------------------------------------------------

# ------------------------------------------------------------------------------
# Question 2.1
# ------------------------------------------------------------------------------

# ------------------------------------------------------------------------------
# Question 2.2
# ------------------------------------------------------------------------------
```