Half-day Workshop on Phylogenetic Comparative Methods

Simon Joly

2023-02-12

Contents

Chapter 1

About

This document consist in an introduction to the comparative methods. It contains theory as well as practical examples in R on Phylogenetic Generalized Least Squares (PGLS). It was developed for a half-day workshop that consists in short presentations followed by R exercises. Note that the present document should pretty much stand by itself because most of the theory given in the presentations are incorporated into the theory sections. Therefore, this document should contain all the necessary information to understand the examples.

I assume that the readers are "reasonably" familiar with R as well as with linear regression and its assumptions. There are a lot of good R introductory tutorials on the web and for linear models. Zuur et al. (**?**) provide a good introduction to linear models, mixed-effects models and model comparison. Good introductions to model fitting in R can also be found on Dolph Schluter's webpage and among the QCBS workshops.

1.1 Useful resources

The links below might provide additional information of interest.

Luke Harmon's book - Phylogenetic Comparative Methods

Course on Phylogenetic Comparative Methods

Introduction to phylogenies in R - pdf

Reading and making phylogenetic trees in R - pdf

1.2 Source

This tutorial is publicly available and is hosted on github in the repository github.com/simjoly/ComparativeMethods-HalfDayWorkshop

1.3 Disclaimer

This tutorial is provided as is, without any guaranty that it will work or that the analyses will be up to date.

Chapter 2

Ahead of the workshop

Here are a few things you should know and you should do ahead of the workshop.

2.1 Install R and the required packages

To perform the examples of this document, you will need to have the R software installed on your computer. I strongly recommend that you install RStudio. Although R Studio is not required, it facilitates interactions between scripts and the R console and provides many great tools.

After installing R, you will have to install some packages. For this specific tutorial, we will need to load the following R packages.

```
library(nlme)
library(ape)
library(RColorBrewer)
library(ggplot2)
```

To execute the code of this tutorial in R, I suggest that you create a new script (File>New File>R Script) where you paste the code copied from the boxes. In R Studio, you can then run this code by selecting the lines you want to execute and then press run (or associated shortcut). This will replicate the analyses presented in the tutorial. You should save the script file in a directory dedicated for the workshop where you will also place the data files required (see section **??**). Then, you should make sure that you script (and data) are in the R working directory. In R Studio, this can be set form the menu: 'Session > Set Working Directory'.

If some of the packages above are not yet installed on your computer, you get error messages when trying to load them. If this is the case, you will have to

install them using the function `install.packages()`. You only have to install them only once.

```
install.packages('nlme')
install.packages('ape')
install.packages('RColorBrewer')
install.packages('ggplot2')
```

Once the packages are installed, you can load the packages using the `library()` function. Also note that if you are using both the packages `nlme` and `ape`, `nlme` should be loaded first. If you don't do this, you might get errors; you could then restart R and start over.

2.2 Downloading the data

The data you will need for this tutorial can be downloaded from this repository: data.zip.

I suggest that you download the folder, uncompress it, and place is in a dedicated folder where you will also save the script with all the commands you will use.

Chapter 3

An introduction to Phylogenetic Comparative Methods

Phylogenetic comparative methods were introduced by Joseph Felsenstein in 1985. The idea of phylogenetic comparative methods was to correct for the non-independence of species in statistical tests because of their shared evolutionary histories. Indeed, two species may look similar not because they live in the same environment but because they are closely related. Consider the following angiosperm phylogeny.

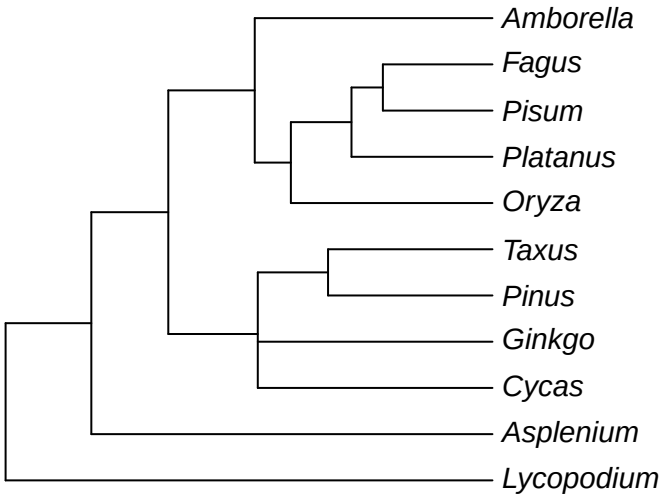
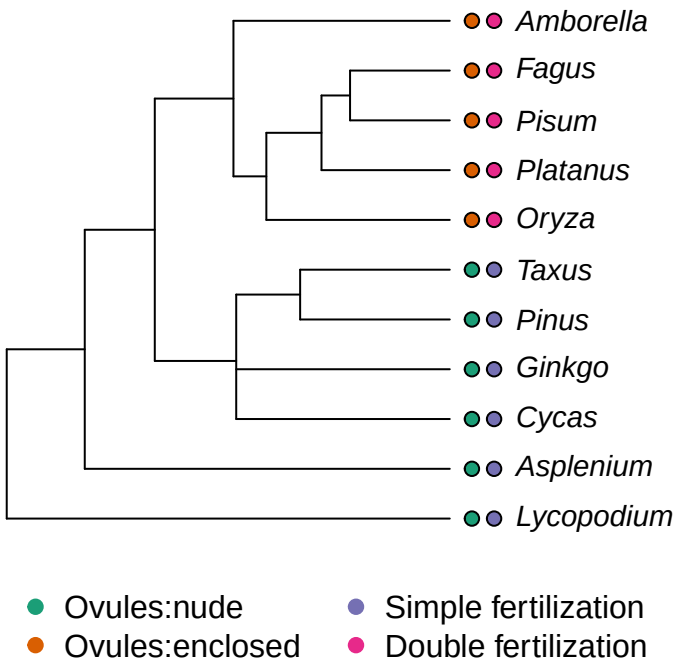


Figure 3.1: land plant phylogeny

It is clear that *Fagus* (beech) and *Pisum* (pea) are more likely to share similar characteristics compared to *Asplenium* (a fern), because they share a more recent common ancestor. In other words, their evolutionary histories are shared over a longer period than with *Asplenium*. As such, they have more chance to have more similar traits (and in fact they do). For instance, take two characters, ovule and fertilization type, within this group.



Ignoring the phylogeny, we might be tempted to see a strong correlation between these two characters. Indeed, the states between the two characters show a perfect correspondence. Using standard contingency table statistics, we could do a Fisher exact test:

```
fisher.test(matrix(c(5,0,0,6),ncol=2))
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  matrix(c(5, 0, 0, 6), ncol = 2)
## p-value = 0.002165
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  2.842809      Inf
## sample estimates:
## odds ratio
##        Inf
```

The test suggests that the assotiation is highly significant. However, we know that the comparisons made are not completely independent. Actually, both characters evolved only once, and this along the same branch.

A more appropriate way to frame the question would be "what is the probability that two characters evolved along the same branch?". This can also be calculated using a contingency table, but this time taking the branches of the phylogeny as the units of observation.

In the example, there are 18 branches and the two characters evolved only once and on the same branch. The contingency table when considering the changes along the branches looks like this:

	Change in trait 2	No change in trait 2
Change in trait 1	1	0
No change in trait 1	0	17

With this table, Fisher's exact test will give the following result:

```
fisher.test(matrix(c(1,0,0,17),ncol=2))
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  matrix(c(1, 0, 0, 17), ncol = 2)
## p-value = 0.05556
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.4358974       Inf
## sample estimates:
## odds ratio
##        Inf
```

You can see that the result is no longer significant.

While this approach for taking into account the phylogenetic relationships is correct, more powerful comparative methods have been developed. One useful and powerful approach is the Phylogenetic Generalized Least Squares (PGLS). But before we introduce PGLS, we do some revision and look briefly at the standard regression.

Chapter 4

The linear regression model

4.1 Theory

The linear model has the following form:

$$\mathbf{y} = \alpha + \beta\mathbf{x} + \mathbf{e}$$

$\mathbf{y}$ is the response (or dependent) variable, $\mathbf{x}$ is the explanatory (or independent) variable, and $\mathbf{e}$ represent the residuals or in other words the variation not explained by the model. For a simple linear regression model, this represents the distance between the observations (i.e., the real data) and the regression line (i.e., the prediction of the model) along the y axis. The parameters α represents the intercept, which is the y value where the regression line crosses the y axis, whereas the parameter β represent the slope of the line. In practice, you take a sample of size N and you get estimates $\hat{\alpha}$ and $\hat{\beta}$ for the intercept and the slope, respectively. When the linear regression is fitted using ordinary least squares (OLS), the residuals $\mathbf{e}$ are assumed to be normally distributed with an expectation 0 and variance σ^2. In mathematical terms, $\mathbf{e} \sim N(0, \sigma^2)$.

Obtaining reliable estimates with a linear regression implies that the data meets several assumptions, amongst which are normality, homogeneity, fixed X, independence, and correct model specification. We won't review all these here, but we will focus on one that is often violated when the data are phylogenetically structured, which is **independence**. This assumption is important as a lack of independence invalidates important tests such as the F-test and the t-test.

You get a violation of independence when the $\mathbf{y}_i$ value at $\mathbf{x}_i$ is influenced by other $\mathbf{x}_i$. Obviously, this can happen with phylogenetically structured data as a response variable is be more likely to react similarly in closely related species because they share many characters by decent. In other words, the y value for

a species in not completely independent from the y value of a closely related species: their y are correlated. We'll illustrate this in an example below.

4.2 Practice

To provide pratical examples in this workshop, we will use a dataset of tree functional traits from the province of Quebec (**?**). The dataset consists in a number of plant functional traits and in a molecular phylogeny built using the plant barcode markers *rbc*L and *mat*K. The dataset you need to run the examples are already in the /data/ folder of the github repository. However, you can also download them by clicking on the links below.

seedplants.tre

seedplants.csv

Before analysing the data, we will start by opening the data and the phylogenetic tree and clean them to keep only the species present in both the tree and the trait table. This is necessary because some additional species were included in the phylogenetic tree analysis.

```
require(ape)
# Open the documents; it assumes that you are in the main directory of the workshop fo
seedplantstree <- read.nexus("./data/seedplants.tre")
seedplantsdata <- read.csv2("./data/seedplants.csv")
# Remove species for which we don't have complete data
seedplantsdata <- na.omit(seedplantsdata)
# Remove species in the tree that are not in the data matrix
species.to.exclude <- seedplantstree$tip.label[!(seedplantstree$tip.label %in% seedplan
seedplantstree <- drop.tip(seedplantstree,species.to.exclude)
# Remove unnecessary object
rm(species.to.exclude)
```

Now, we can have a look at the data, and then order the plant trait to be in the same order as the species in the tree.

```
# Here is what the loaded data looks like
head(seedplantsdata)
```

```
##   Code       Species.name Occurrence maxH   Wd    Sm Shade    N
## 1 ABBA     Abies balsamea       7759   25 0.34   7.6   5.0 1.66
## 2 ACNE       Acer negundo          0   20 0.44  34.0   3.5 2.50
## 3 ACNI        Acer nigrum          1   30 0.52  65.0   3.0 1.83
## 4 ACPE Acer pensylvanicum        665   10 0.44  41.0   3.5 2.22
## 5 ACPL   Acer platanoides          0   15 0.51 172.0   4.2 1.99
## 6 ACRU        Acer rubrum       3669   25 0.49  20.0   3.4 1.91
```

```
# Order the tree to make it nicer when plotting
seedplantstree <- ladderize(seedplantstree, right = FALSE)
# Name the rows of the data.frame with the species codes used as tree labels
#  and remove the obsolete column with species codes.
rownames(seedplantsdata) <- seedplantsdata$Code
seedplantsdata <- seedplantsdata[,-1]
# Order the data in the same order as the tip.label of the tree. In the present
#  example, this was already the case, but it is an important step for
#  any analysis.
seedplantsdata <- seedplantsdata[seedplantstree$tip.label,]
```

Now that the data is ready, let's fit a linear model and try to explain shade tolerance (Shade) of trees using wood density (Wd). In R, a very simple way to do a regression is to use the function 'lm', which stands for linear model. To fit a linear model, you need to tell the `lm` function which variable is the response variable and which one is the explanatory variable. This is done using formulas in the form `Shade ~ Wd`. The variable at the left of the tilde ('~') is the response variable (`Shade`) whereas the explanatory variale (1 or more) are at the right of the tilde.

```
# Fit a linear model using Ordinary Least Squares (OLS)
shade.lm <- lm(Shade ~ Wd, data = seedplantsdata)
# Print the results
summary(shade.lm)
```

```
##
## Call:
## lm(formula = Shade ~ Wd, data = seedplantsdata)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.87120 -1.02501  0.05628  0.70132  2.38261
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.0010     0.7501   2.668    0.010 *
## Wd            1.8130     1.5676   1.157    0.252
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.146 on 55 degrees of freedom
## Multiple R-squared:  0.02374,    Adjusted R-squared:  0.005992
## F-statistic: 1.338 on 1 and 55 DF,  p-value: 0.2525
```

You can see that the slope estimate (here the parameter `Wd`) is 1.81 and that is not significant (p=0.252). The standard descriptive plots obtained with `plot(shade.lm)` show that there is slightly greater variation in the residuals