Comparative Methods Workshops

Chris Mitchell

Contents

Chapter 1

Welcome

Welcome to the online support materials for the Comparative Research Group at the University of Liverpool. The CRG is made up of staff and students engaging in comparative research across various areas of evolutionary biology.

1.1 For students

The materials here are intended to support you through your LIFE363 honours project. For this project you will be performing a comparative study (see **chapter 1** for more information) on an area of your choosing. At first, this is a daunting task but developing your own research here is excellent experience and gives you the opportunity to research an area that really interests you.

The vast majority of statistics here are performed in R [R Core Team, 2019]. You were introduced to R in LIFE223 as a powerful and flexible tool for statistical analysis. **Chapter 2** contains a brief refresher on some of the basics of R in case you need it. For more detailed recaps, please revisit your materials from LIFE223 as some of the code you wrote is likely to be useful this year!

Throughout this book you will see examples of R code and output like this.

```r
print(hidden.message)
```

```
[1] "Forty-two"
```

The code can be copied and pasted into your own version of R as you see fit. However, I would recommend that for the first time you are using a piece of code, type it out for yourself. This will help you get to grips with what each argument means.

You will also see some interactive R windows where you can enter your code directly into this book and an online version of R will run it. This should give you an opportunity to learn more complex things and develop your R skills dramatically.

The rest of the book is populated with workshops and materials to help you learn specific comparative statistical methods. Some of these will be extensions of what you already met in LIFE223. **Chapter 6** looks at phylogenetically controlled ANOVA and **chapter 14** is all about phylogenetic regression.

Other methods may be entirely new to you such as ancestral state reconstruction (**chapters 7 - 10**) or path analysis (**chapter 15**). Don't be intimidated by this. All the code you need is gathered here and will remain available as long as you need it.

Chapter 2

Introduction

This chapter contains a very brief overview of the research we do in the **Comparative Research Group**. Taxonomically, the work done by group members is extremely broad. We've had projects on primates, octopuses, domestic mammals, birds and more! Here is a sample of titles from previous students.

- Identification of a cognitive niche in benthic octopods and possible areas for future study on cephalopod intelligence.
- Evolutionary precursors for the domestication of Artiodactyla.
- You are not what you eat: Lack of morphological convergence in beak and body size between the nectarivorous avian families Trochilidae and Meliphagidae.
- Investigating how lifestyle factors affect lifespan in reptiles.
- Ecological processes causing encephalisation in Madagascan lemurs.

2.1 What is the comparative method?

The comparative method is a catch-all term for a suite of approaches that involve using comparisons to answer scientific questions. In evolutionary biology, the comparative method refers to making comparisons between species or populations in order to identify patterns and relationships between traits of interest. Used correctly, this approach can be very powerful and allows us to ask large-scale questions about evolutionary patterns, adaptive processes and coevolutionary relationships.

The most basic kind of comparative study is comparing one species or lineage to another. For example, a recent paper made waves in the paleontology community by demostrating (after years of debate) that *Spinosaurus aegypticus* lived an aquatic lifestyle [Ibrahim et al., 2020]. The analysis centered around some newly recovered tail vertebrae with extremely long (1m!) spines. The tail of

Spinosaurus was compared to other animals including terrestrial theropods like *Allosaurus* and semi-aquatic tetratpods such as the crocodile. This comparison showed that the *Spinosaurus* tail was indeed specialised for powerful propulsion through the water (like a crocodile), seemingly settling the debate over whether any non-avian dinosaurs invaded the water.

Other comparative studies take data gathered from many species and search for patterns within that group. Studies like this rely a great deal on work done by others. For example, Simon Reader and colleagues [2011] carried out an extensive literature search looking for examples of five behavioural traits in many species of primate in over 4000 articles published over 75 years. The resulting database included examples of innovation, social learning, tool use, extractive foraging and tactical deception and was used to demonstrate a correlation between these behaviours and brain size, providing evidence of a general intelligence factor in primates similar to that in humans.

2.2 Tree thinking

Comparative studies can be great but there is a problem. In LIFE223 you learned about statistical assumptions. One of the most common and important assumptions of most statistical tests is that data are independent. To run a good comparative study we need to know that the data points we have are independent of each other. In evolutionary biology, we know that this isn't the case!

All living things exhibit a pattern of relatedness which depends on how much shared evolutionary history they have. For example, chimpanzees and human beings diverged about 6-7 million years ago. This means that they have much more shared evolutionary history than chimpanzees and *Spinosaurus* which are separated by hundreds of millions of years.

The best way of visualising this pattern of relatedness is with a phylogenetic tree.

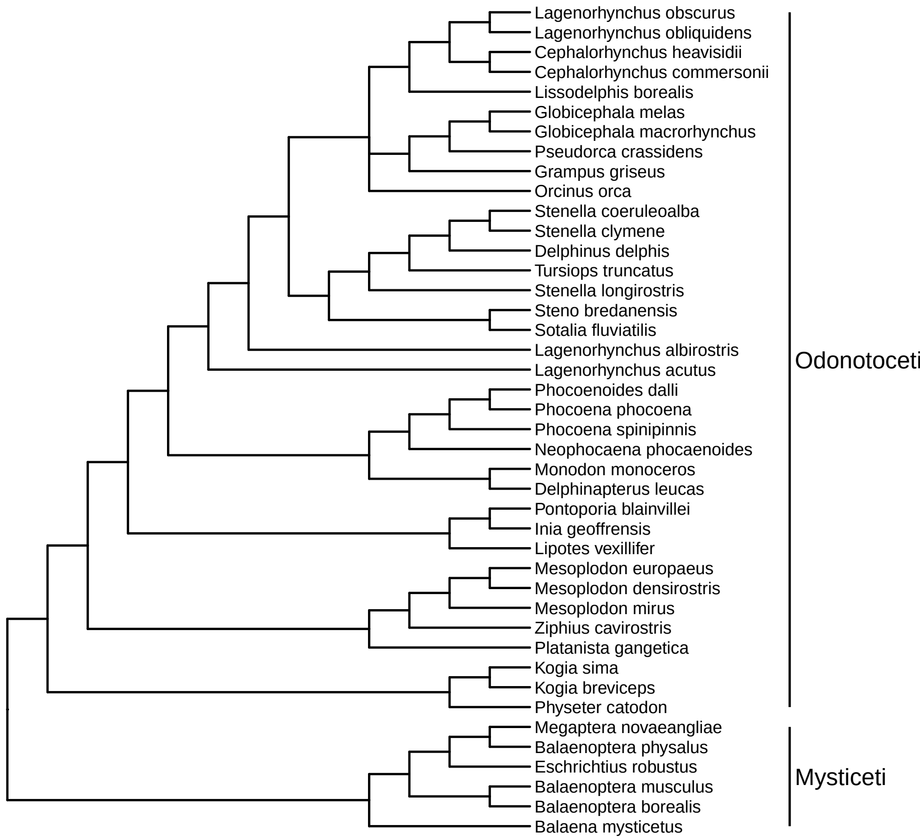


Figure 2.1: A cladogram of 42 cetacean species.

The extant species are displayed on the **tips** of the tree and are connected to each other according to the degree of relatedness by the **branches**. Figure 2.1 shows us the pattern of relatedness of 42 cetacean species. If we wanted to use these species in a comparative study to investigate the evolutionary history of the group, we would not have independent data points. This means that the assumptions of most statistical tests would be violated and we couldn't trust the results!

This is where phylogenetics comes to the rescue. We can use the pattern of relatedness described by the phylogeny to control for the non-indepedence of data points. To show you what I mean, let's consider body size in those 42 species of cetacean. If we were to show the distribution of body size in the group, we would see that the vast majority of the largest sizes are found in the mysticetes whereas the smaller species tend to be odontocetes. If we viewed these data points as all independent we might say that very large bodies have evolved 7 times in the group (once for each mysticete and once for the sperm whale) whilst small body size has evolved in all the other species (35 times).

In fact, the close relatedness of 6 of the large bodied species suggests that large body size evolved once and not independently for each of these species. Their shared evolutionary history explains why their traits (body size in this case) are so similar. The seventh example of a large body (sperm whales) does not share very much history with the other 6 and this may be of some interest to us. It suggests an independent evolution of large body size and potentially something of interest to us as researchers.

So hopefully you can see how taking phylogeny into account can be illuminating. For a broader (and much more useful) introduction to phylogenetics and its use in evolutionary biology, check out these sources:

- Tree Thinking: An Introduction to Phylogenetic Biology [Baum and Smith, 2012]
- The Comparative Approach in Evolutionary Anthropology and Biology [Nunn, 2011]

Chapter 3

R recap

In LIFE223, we taught you how to use R for statistical analysis and visualising data. This chapter will contain a basic overview of some of the things from 223 that you may find useful as we proceed. You only need to bother with this if you are new to R or have blocked it from your memory since you last used it.

3.1 Basics

R works well as a calculator.

```
6*7
```

```
[1] 42
```

However, R is capable of a great deal more than just simple mathematical operations like multiply and divide. It also has functions that can calculate some common descriptive stats like mean and standard deviation.

```
mean(x)
```

```
[1] 41.94244
```

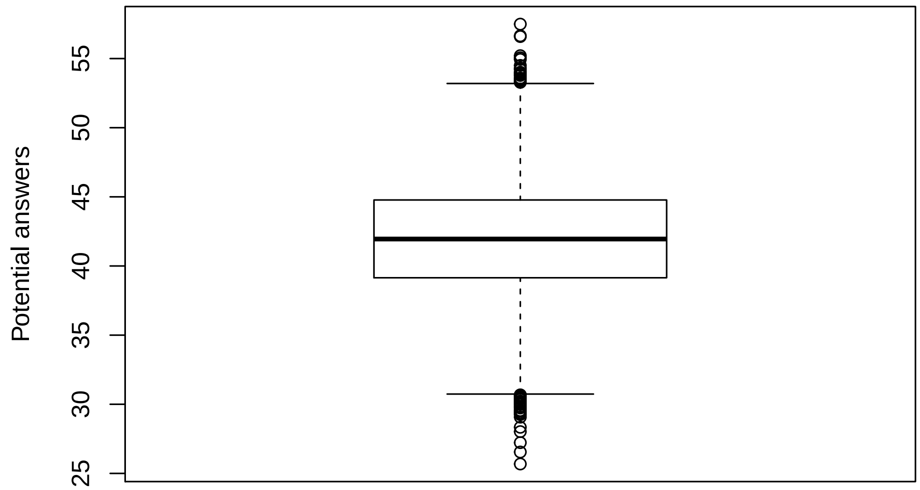
```
sd(x)
```

```
[1] 4.190698
```

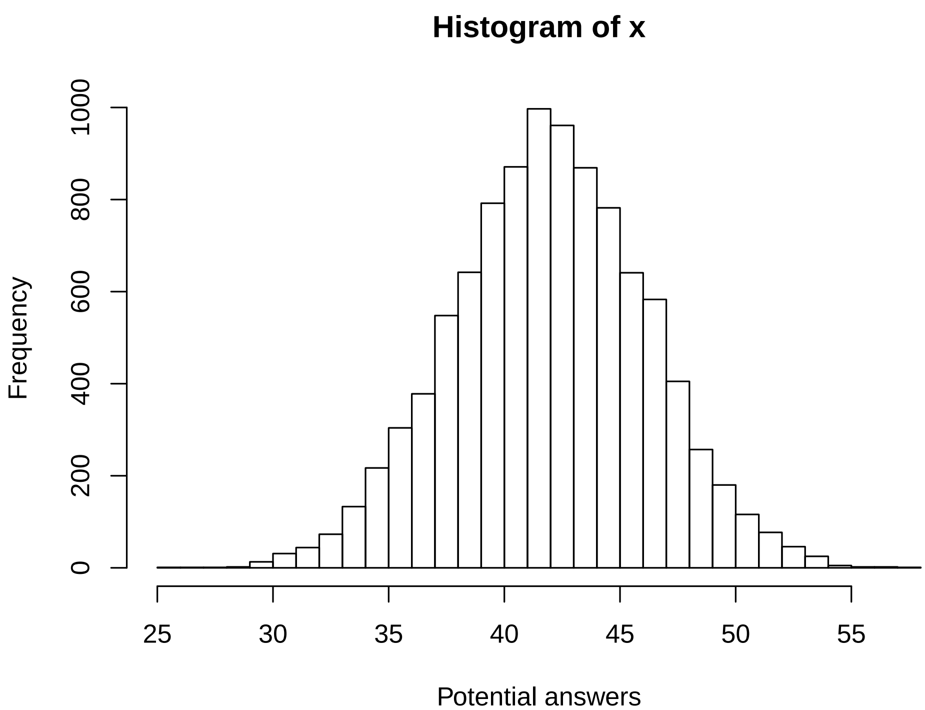
3.1.1 Plotting

R is also a very flexible graphical tool. From LIFE223, you probably remember a few basic plotting functions. Each function in R has arguments that can be added to label axes or change point size as you can see in these plots.

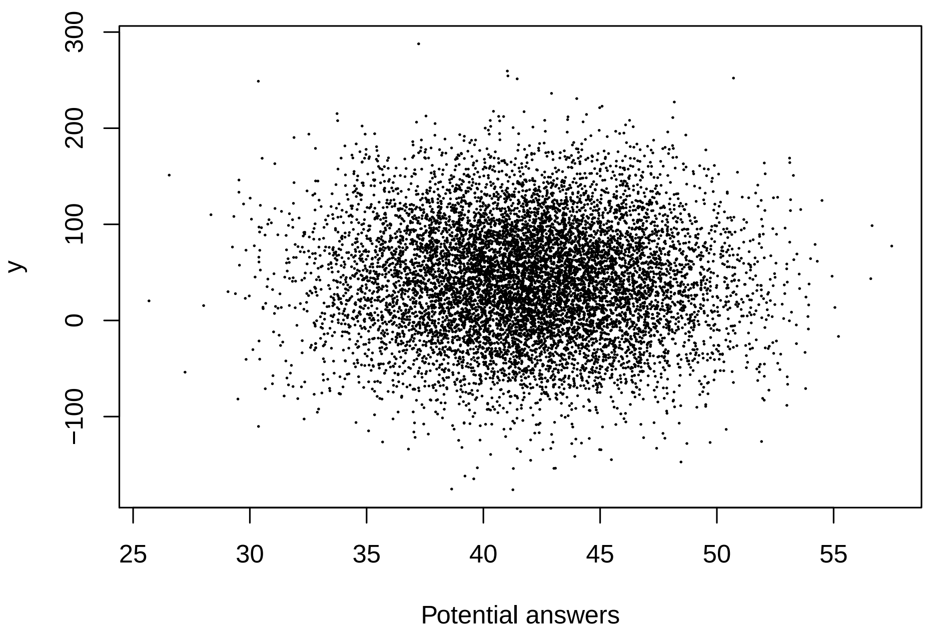
```
boxplot(x, ylab = "Potential answers")
```



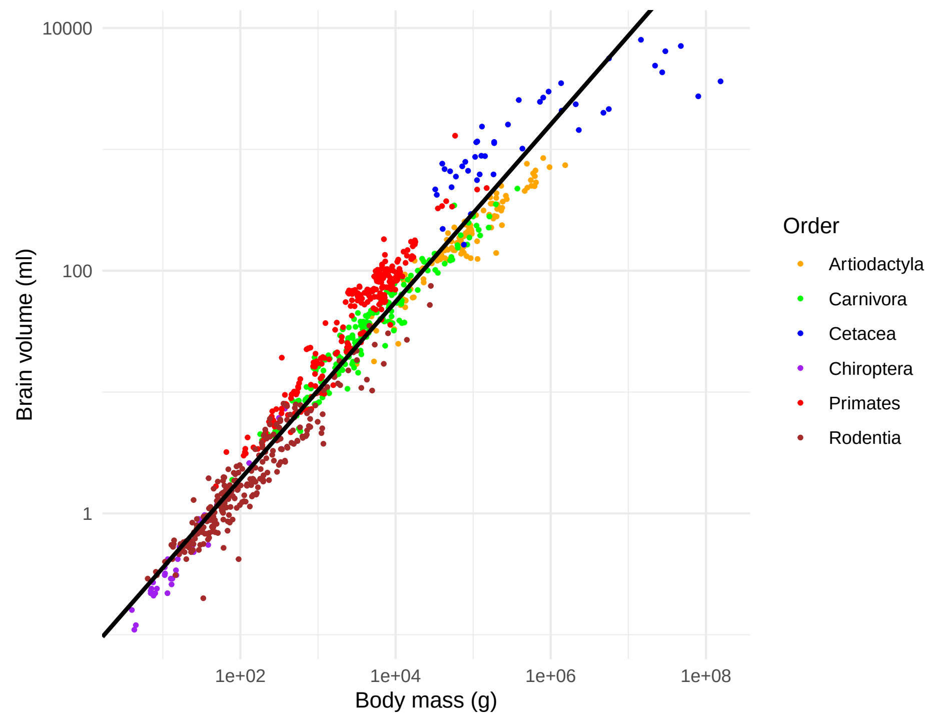

```
hist(x, xlab = "Potential answers", breaks = 25)
```



```
plot(x, y, xlab = "Potential answers", pch = 19, cex = 0.1)
```



For much of this book, I will actually be doing most plotting in a package called **ggplot2**. This package has a slightly different syntax to get used to but the increased flexibility you have will be a good payoff. Plus the plots look quite nice.



3.1.2 The working directory

The working directory is the folder on your computer where R's attention is focused. This is where you should store any files you need R to open. You can find out the path of the current working directory using the function **getwd()**

```
getwd()
```

```
[1] "/Users/chrismitchell/Google Drive/University of Liverpool/GitHub Stuff/bookdownCR
```

If this isn't the folder we want as our working directory, we can just as easily change it with **setwd()**

```
setwd("~/Desktop/My R Folder")
```

If you are using RStudio there is also a shortcut to do this in the *Files* pane (usually bottom right). Use this pane to navigate to your chosen directory and then use the drop down menu under *More* (look for a blue cog) to set the current folder as your working directory.

If you aren't using RStudio, I'd strongly suggest you start. It's much more user friendly than base R.

3.1.3 Loading data

Data comes in many forms and R is capable of reading most of them if you know the correct functions. One of the most common formats is *comma separated values*. This has the file extension .csv at the end of the filename. If you open a .csv file with MS Excel or Numbers, you will see that it usually looks much like a spreadsheet. To load a .csv data file into R, use the function **read.csv()** as shown here.

```
data <- read.csv(file = "DATAFILE.csv")
```

For other data formats, you may require a different function. For example, data may be provided as a text file (extension .txt). In this case, you need **read.table()**. Note that with this function you need to specify that your data has a header (a top row with names for columns) whereas **read.csv()** assumes this by default.

```
data <- read.table(file = "DATAFILE.txt", header = TRUE)
```

3.1.4 Subsetting

Let's say I want to subset my data based on a certain condition. I can achieve this multiple ways but one of the simplest is the function subset.

```
newdata <- subset(data, species == "Homo sapiens")
```

This function takes a subset of the object *data* and applies the rule that the value of each row in the column *species* be *Homo sapiens*. Thus it extracts the lines of data that are from human beings.

3.2 Errors

Error messages are a part of life with R. You are not expected to be able to interpret every single one immediately and you definitely shouldn't panic or give up when you get one.

Here's a basic error message:

```
data <- read.csv("mydaat.csv")
```

```
Warning in file(file, "rt"): cannot open file 'mydaat.csv': No such file or
directory
```

```
Error in file(file, "rt"): cannot open the connection
```

The message tells me that R "cannot open the connection" and no such file exists. This means that R cannot find the file I was looking for in the current working directory. It could be because I haven't set the correct working directory or the file is there but in a different format. In this case, the error has appeared because I have spelled the name incorrectly. I have sent R looking for a file called *mydaat.csv* instead of *mydata.csv*. Always remember that R is a useful idiot and will only do exactly what you tell it to do!

3.3 Google

The most important skill you need for using R is the ability to use Google (other search engines are available). It may seem odd but almost any problem you will ever encounter with R can be solved by a quick Google search.

If you come up against a confusing error message, copy and paste the message into Google. You will quickly land on one of the forums where someone else has asked about the same error message. The odds are pretty good you'll discover an explanation for the problem there.

If you don't know how to do something, pop the name of what you want to do into Google and add "in R" at the end and there will almost always be a tutorial on the first page of results with exactly what you need.

Seriously, Google is your strongest ally here. The community of R users has populated the internet with endless advice and guidance for every level from beginner to the most advanced of users. That brings me to my next point…

3.4 Stealing

If imitation is the greatest form of flattery then learning to code in R is just about the most flattering thing you can do. The internet is teeming with examples of R code for all kinds of purposes including in this very book. Take it without thinking twice.

You will have acheived a pretty good level of skill in R when you can take someone else's code and edit it for your own purposes. This is the **core skill** of R and once you can do that, you'll be unstoppable.

Chapter 4

Phylogenetic trees and where to find them

This chapter is a brief overview of some key concepts that may be useful when performing comparative research.

4.1 Phylogeny

Phylogeny is the term used to describe the evolutionary history of a group of species. The most common representation of phylogeny is a phylogenetic tree. There is a lot of terminology around phylogenetic trees. Here we will start with the very basics that will come up a lot in this book.

The **tips** of the tree represent the species/populations/individuals described by the tree. The **branches** of the tree represent the pattern of relationships between species. The **nodes** of a tree represent the most recent common ancestor of the lineages that diverge from that node. A **clade** is a monophyletic grouping of lineages. A grouping is **monophyletic** only if all members of that group descend from a common ancestor to the exclusion of others. for example, humans and apes form a monophyletic grouping but humans, apes and parrots do not.

Here is an example of a phylogenetic tree displaying the relationships of modern dog breeds taken from a nice paper investigating the evolutionary history of the domestic dog [Parker et al., 2017]

The tree shown above is a **cladogram** meaning that the lengths of the branches do not carry any real meaning. This tree is only useful for interpreting the relatedness of the species. It does not give us any information about the amount of evolutionary change or the amount time between nodes.

By contrast, the following tree has **branch lengths**. The tree is based on genetic analysis of 173 species of hymenoptera (bees, ants and wasps) [Peters et al., 2017]. In this tree, the branch lengths represent time (in millions of years) as calculated from analysis of over 3,000 genes and calibrated using fossils.

Branch lengths do not always represent evolutionary distance as time. In some cases, evolutionary distance is represented as the amount of change on each branch. The next tree was built based on a brain development gene (MCPH1) in cetaceans (whales, dolphins and porpoises) [McGowen et al., 2011]. On a tree like this, longer branches indicate more character changes along the branch. In this case the character changes will be changes in genetic sequence but for other trees it may be morphological characters, protein sequences characters or a combination.

4.2 Building trees

Developments in the field of phylogenetics have meant that there are many ways to construct a phylogeny. Many of the modern methods are highly sophisticated and for now, these are not the subject of this book. However, it may help you to have a brief introduction to the logic behind building a phylogeny.

4.3 Locating trees

4.3.1 File format

Chapter 5

Plotting trees in R

5.1 Phylogenies in R

From LIFE223, you know R as a powerful statistical tool. You will also be aware that it is an incredibly flexible tool for plotting data. In this workshop, we will be working with phylogenies in R and manipulating them to produce informative plots.

5.1.1 Packages used

In this section we'll mostly be using a package called ggtree [Yu et al., 2017, 2018]. To install it, we need another package called **BiocManager** [Morgan, 2019].

```
install.packages("BiocManager")
BiocManager::install("ggtree")
library(ggtree)
```

```
library(ggtree)
```

We will also need to use phylobase [R Hackathon et al., 2019], ggimage [Yu, 2019] and it would help to have the tidyverse packages loaded [Wickham, 2017] since we'll be using the syntax of ggplot2. If you get an error message, make sure the packages are installed first.

```
library(tidyverse)
library(phylobase)
library(ggimage)
```

5.2 Importing your tree

Let's start by importing a tree. Make sure your working directory is set to wherever you have saved the tree_newick file. If you run this line, you should see an object called "tree" appear in your global environment.

```
tree <- read.tree("tree_newick.nwk")
```

If we take a look at the structure of our tree object using the **str** function we can see that the tree is stored as an object of class **phylo**. If you are using a block of trees (more on this subsequent chapters) it will be an object of class **multiphylo**.

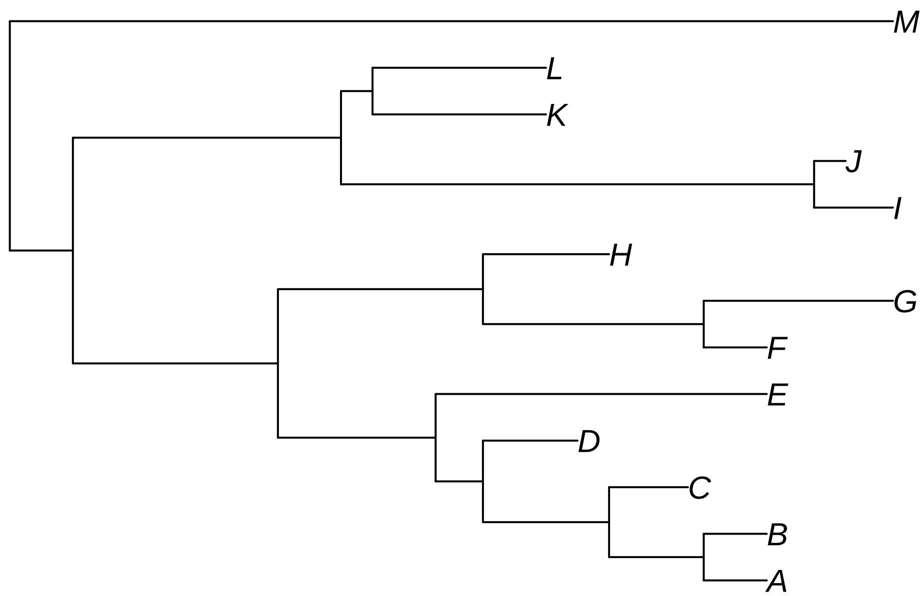
```
str(tree)
```

```
List of 4
 $ edge       : int [1:24, 1:2] 14 15 16 17 18 19 20 20 19 18 ...
 $ edge.length: num [1:24] 4 13 10 3 8 6 4 4 5 6 ...
 $ Nnode      : int 12
 $ tip.label  : chr [1:13] "A" "B" "C" "D" ...
 - attr(*, "class")= chr "phylo"
 - attr(*, "order")= chr "cladewise"
```

We can see a list of 4 elements of the tree object. The first (**edge**) contains the edges (also known as branches) of the phylogeny and their labels. The next is **edge.length** which contains the lengths of the branches if present (see **chapter 3** for more details). **Nnode** specifies the number of nodes and finally **tip.label** contains the labels of the tips. In this case, we just have letters for tip labels.

Things are often clearer when we plot them. We can do this for trees with the **plot** function in base R. This function is incredibly versatile and you should recognise it from LIFE223. Here we are using very different arguments.

```
plot(tree)
```



This plot is fine for a quick check to make sure the tree looks as we expected it to. Let's look at making a more attractive plot with ggtree.

5.3 ggtree

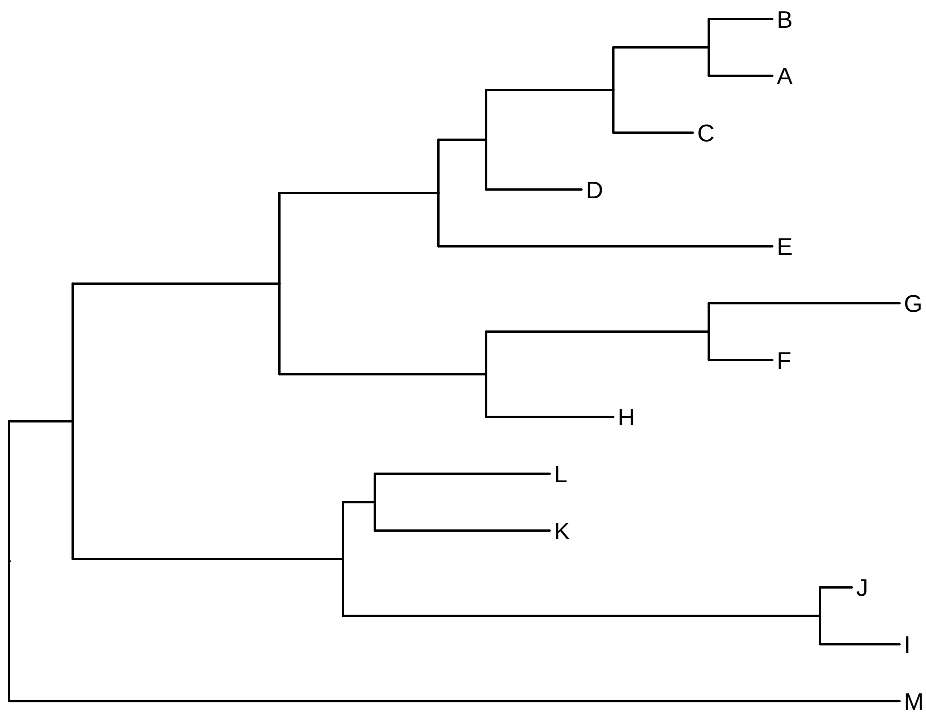
The package ggtree is an extension of **ggplot2**, a popular plotting package from the **tidyverse** family of packages. The syntax we'll be using here is a little different that what you may be used to so don't get intimidated. **ggtree** uses the same syntax as **ggplot2**. This works by creating layers (known as **geoms**) and plotting them over each other to build up the plot.

We'll start by using ggtree to plot our tree. Below is the base layer of the plot. There are many other options we can include to customise our tree. Try some out in this R window to see how they effect your plot.

5.3.1 Geoms

Geoms are new layers to plot on or alongside your tree. Now let's try plotting it whilst adding new layers. These geoms can be combined as you see fit. This gives you a lot of flexibility in how you plot your trees. For example, we can add a geom to include the tip labels for our tree.

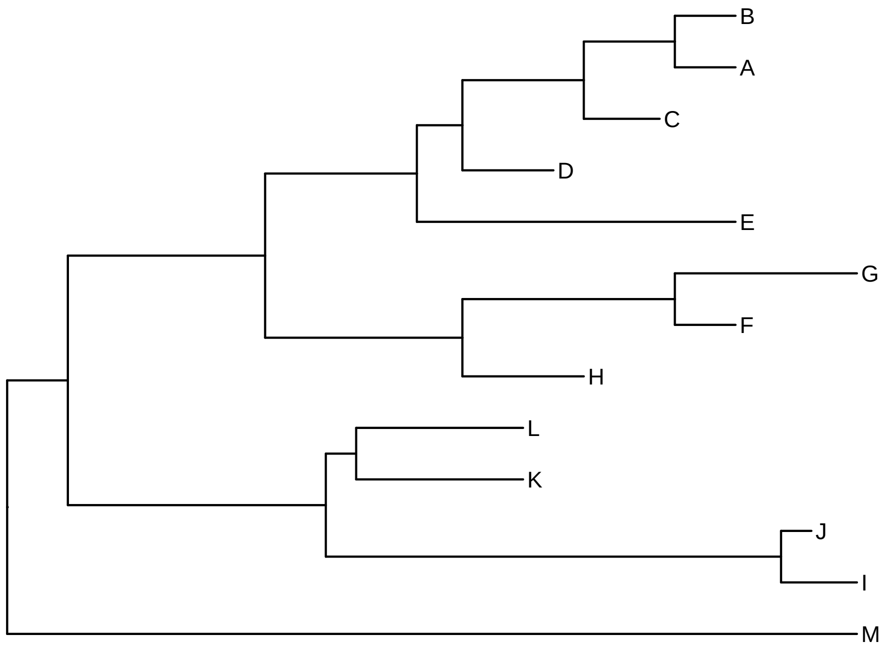
```
ggtree(tree) +
  geom_tiplab()
```



And we can add a title

```
ggtree(tree) +
  geom_tiplab() +
  ggtitle("A phylogeny of letters. For some reason...")
```

A phylogeny of letters. For some reason...



There are many geoms you can use to add more information to your plot. Here are just a few that you may want to investigate.

```
geom_tiplab() #adds tiplables
geom_tippoint() #adds points at the tips
geom_nodepoint() #adds points at the nodes
geom_nodelab() #adds labels for nodes
geom_cladelabel() #adds labels for clades
```

5.3.2 Labelling clades

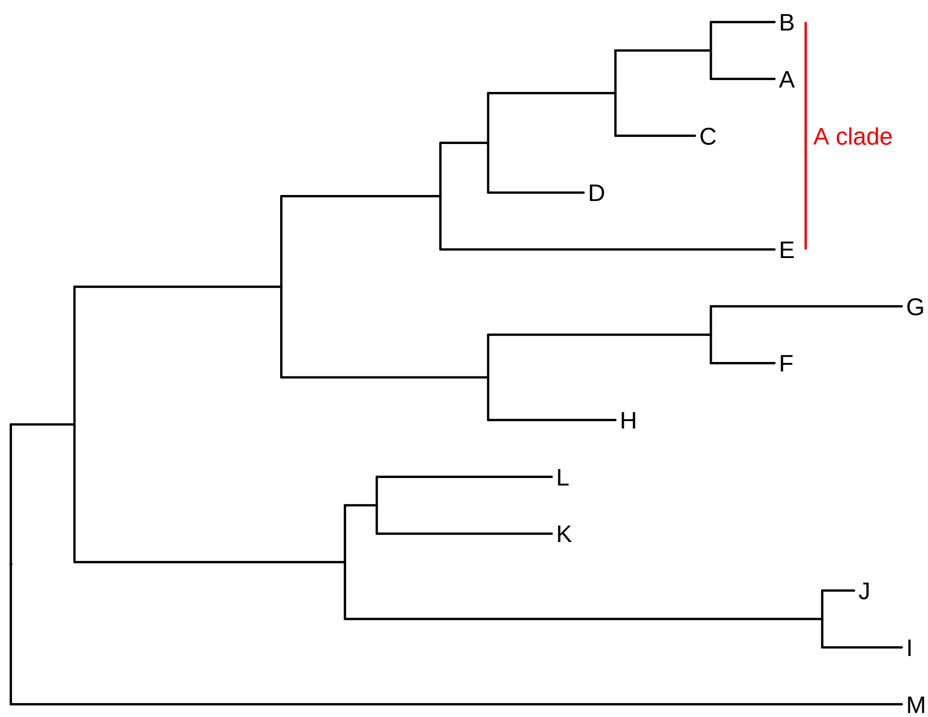
As an example of what you might like to do with ggtree, let's have a look at adding some labels to identify some clades on our tree. To label clades, we need to be able to identify the node of the most recent common ancestor. The function **MRCA** in the package **phylobase** [R Hackathon et al., 2019] tells us that the common ancestor of C and E is node 17.

```
phylobase::MRCA(tree, tip = c("C", "E"))
```

```
[1] 17
```

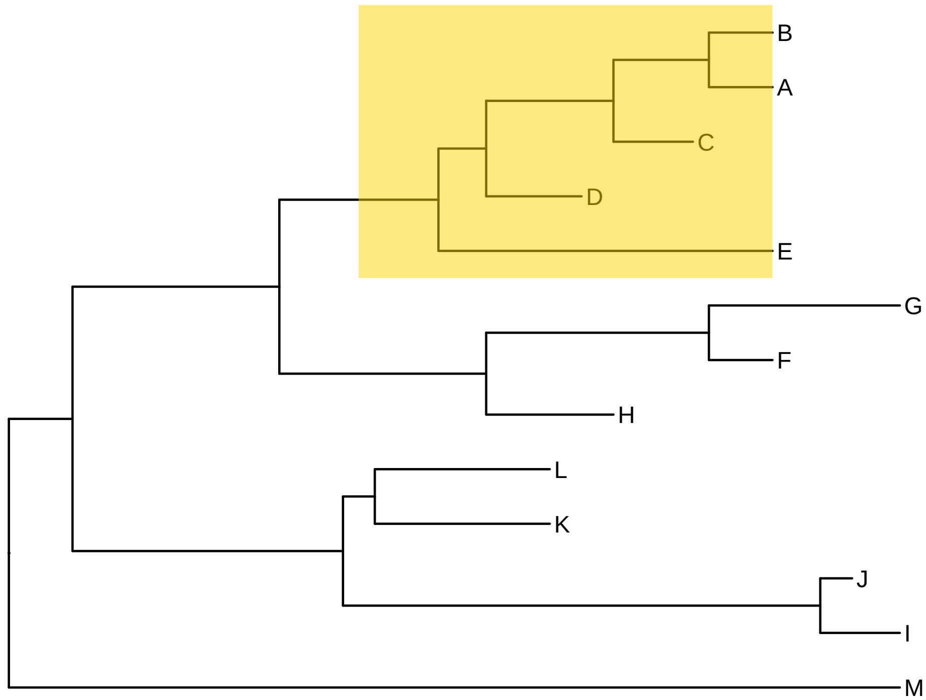
We can now use the **geom_cladelabel** geom to add a simple label for the clade descended from the appropriate node. Take note of the arguments I've added to customise the geom. You may want to play around with these options yourself to see how they work.

```
ggtree(tree) +
  geom_tiplab() +
  geom_cladelabel(node=17, label="A clade",
                  color="red2", offset=1)
```



Pretty good but there are other options. This is a matter of personal preference. You may prefer to overlay a translucent rectangle over your clade of interest.

```
ggtree(tree) +
  geom_tiplab() +
  geom_hilight(node=17, fill="gold")
```



Use the R window below to experiment with the available geoms in ggtree. Find a combination that suits you and your tree.

5.4 Adding images to trees

As you probably noted in chapter 3, adding images to a plot is an excellent way to annotate your tree. The ggtree package can do this as you can see here.

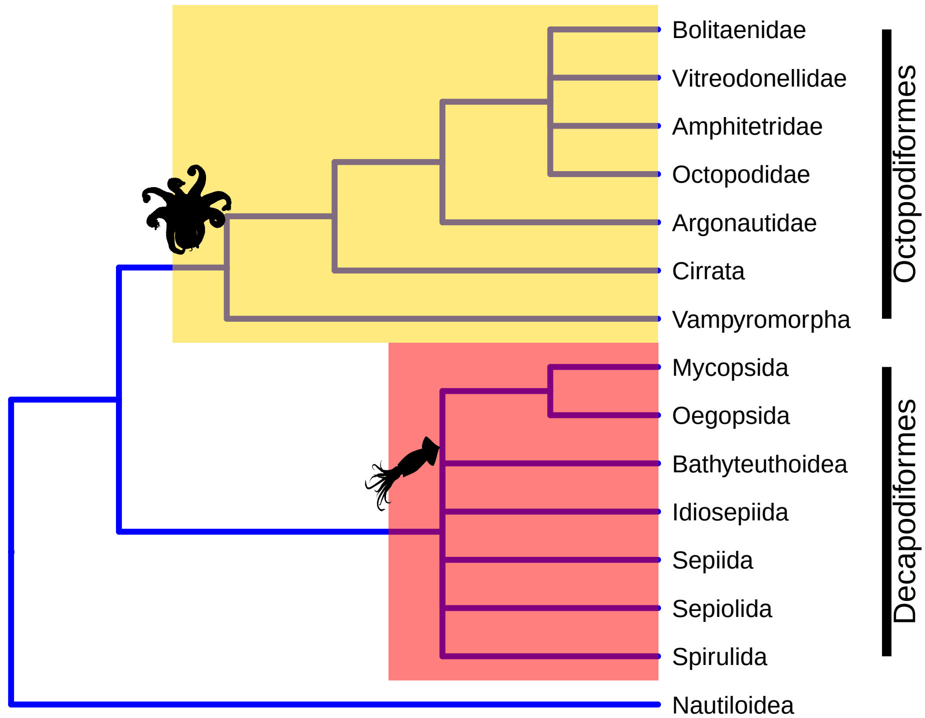


Figure 5.1: Plot of cephalopod families annotated using ggtree and Phylopic.

This phylogeny is annotated in a number of useful ways. The tip labels describe cephalopod families. The superorders (octopodiformes and decapodiformes) are highlighted by gold and red rectangles as well as a bar across the tips. this demonstrates how multiple geoms can combine to make a plot easy to interpret.

The most interesting thing for our purposes are the silhouettes at the root of each superorder. The octopodiformes have an octopus and the decapodiformes have a squid as example taxa from within the superorder.

5.4.1 Phylopic

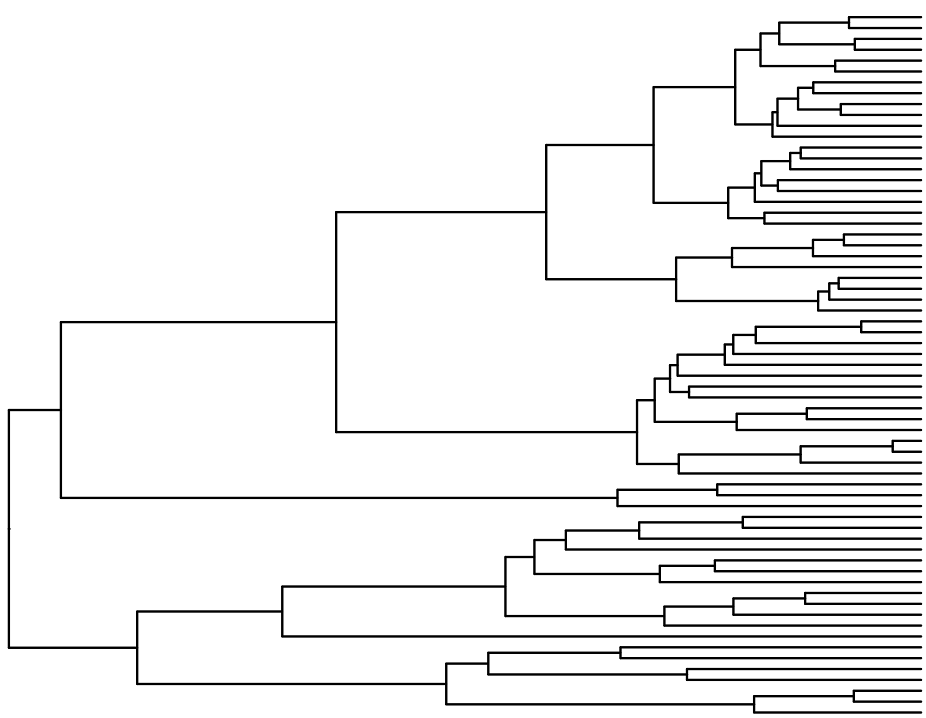
The silhouettes used for that plot are from a website called Phylopic. Phylopic provides open source biological silhouettes that are free to use. We're now going to look at how to get these silhouettes and use them to annotate our trees.

Let's start with loading an example tree. This one is a primate tree courtesy of Randi Griffin. You'll notice that I'm loading this tree using a url. This is because I'm loading a file directly from GitHub, a repository for all sorts of code and the host of this site! Randi (and many other coders) make some of the things they produce freely available through GitHub. This can be data, files or code.

```
primates <- read.nexus("https://raw.githubusercontent.com/rgriff23/Dissertation/master/Chapter_2/
```

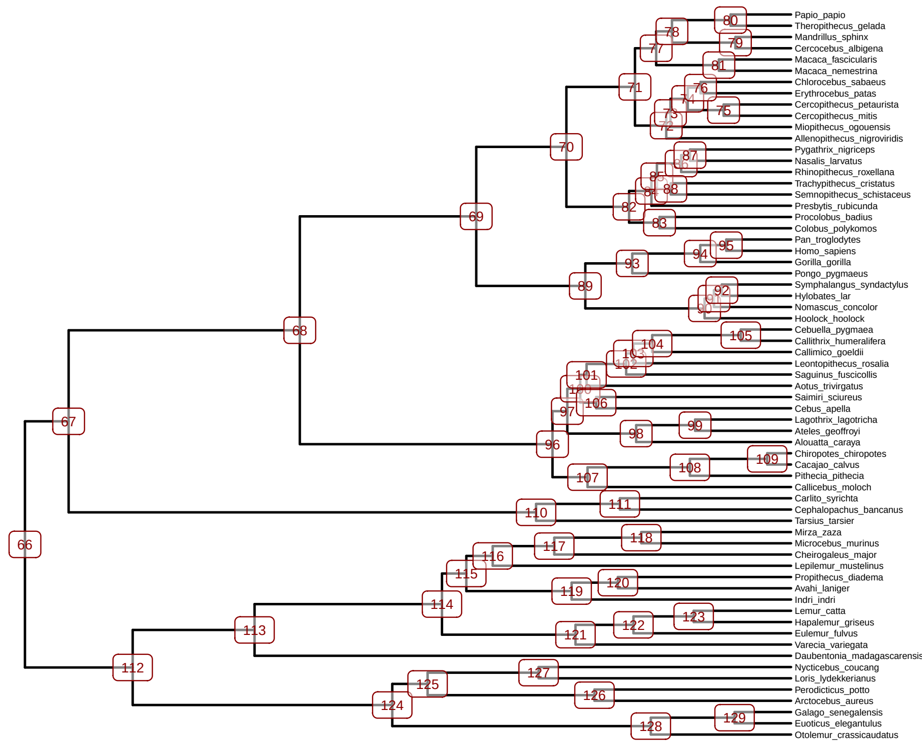
Let's plot the new tree first. Here I'm assigning the plot to a named object (p1) in R. This means that instead of immediately printing out the plot, R stores it in the working directory. The reason for doing this will become clear as we go on. It saves us typing out every line of code each time we want to add a new geom!

```
p1 <- ggtree(primates)
p1
```



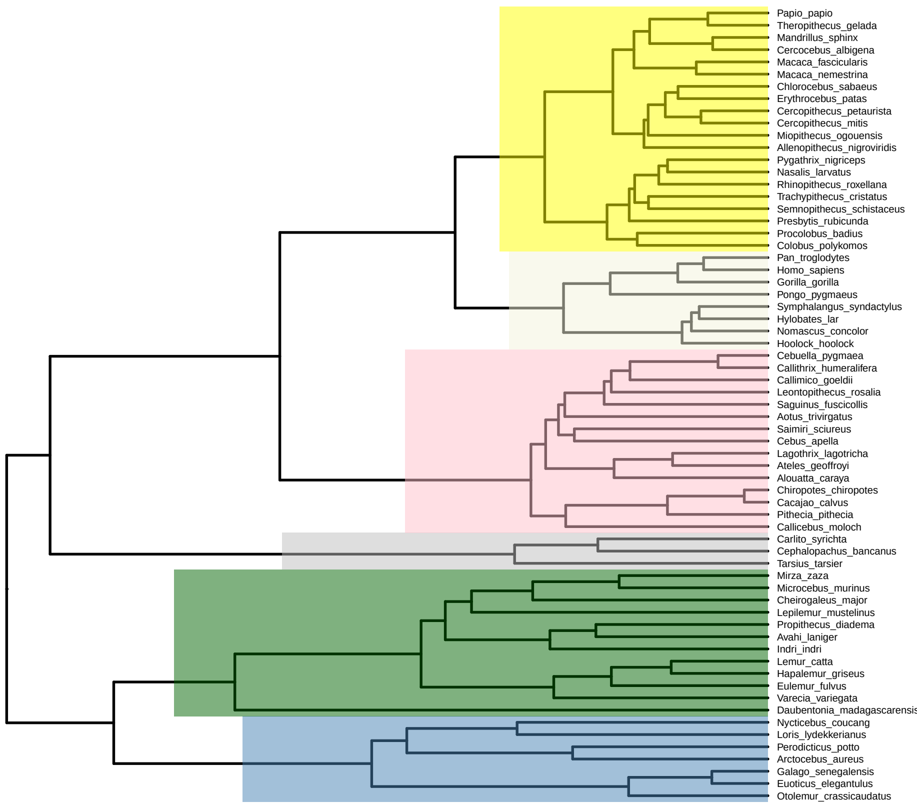
Let's use what we already know about ggtree to customise this plot into something more useful. In particular, this plot is quite useful because it tells us the numbers of each node and we will need that later on.

```
ggtree(primates) +
   xlim(0,90) +
   geom_tiplab(size=1.5) +
   geom_label2(aes(subset=!isTip, label=node), size=2, color="darkred", alpha=0.5)
```



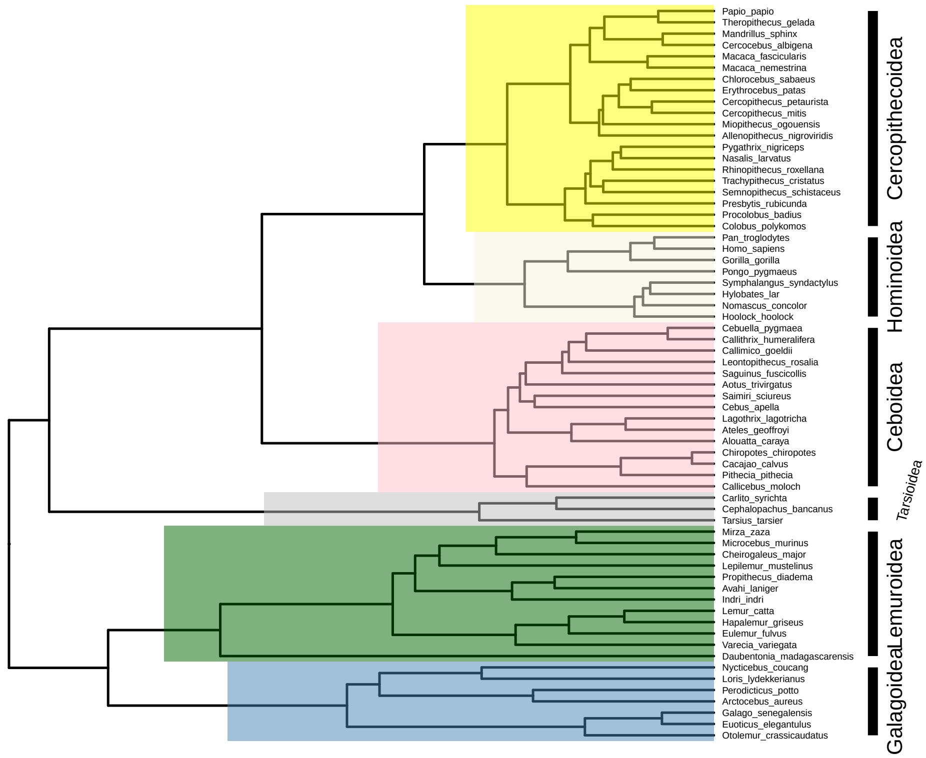
Let's label the 6 primate superfamilies using the node numbers I have extracted from the previous plot. You can choose whatever colours you prefer here. I've also added some useful features to this code. the use of **xlim()** can be very useful when plotting a tree with some extra space for more details. Here I've set the limits of the x dimension (the horizontal) to be between 0 and 100. This gives me space for later annotations.

```
p2 <- ggtree(primates) +
  xlim(0,100) +
  geom_tiplab(size=1.5, offset=0.5) +
  geom_hilight(node=124, fill="steelblue", alpha=0.5) +
  geom_hilight(node=113, fill="darkgreen", alpha=0.5) +
  geom_hilight(node=110, fill="gray", alpha=0.5) +
  geom_hilight(node=96, fill="pink", alpha=0.5) +
  geom_hilight(node=89, fill="beige", alpha=0.5) +
  geom_hilight(node=70, fill="yellow", alpha=0.5)
p2
```



So far so good. Let's add on bars to label the superfamilies like I did for the cephalopod version. This time, I'll add the new details to the object p3 to save retyping. Take note of the arguments in each label. You may need to twist these with some trial-and-error to make sure they suit your plot window.

```
p3 <- p2 +
  geom_cladelabel(124, "Galagoidea", offset=15, barsize=2, angle=90,
                  offset.text=1.5, hjust=0.5, fontsize=3) +
  geom_cladelabel(113, "Lemuroidea", offset=15, barsize=2, angle=90,
                  offset.text=1.5, hjust=0.5, fontsize=3) +
  geom_cladelabel(110, "Tarsioidea", offset=15, barsize=2, angle=75,
                  offset.text=2.5, hjust=0.2, fontsize=2) +
  geom_cladelabel(96, "Ceboidea", offset=15, barsize=2, angle=90,
                  offset.text=1.5, hjust=0.5, fontsize=3) +
  geom_cladelabel(89, "Hominoidea", offset=15, barsize=2, angle=90,
                  offset.text=1.5, hjust=0.5, fontsize=3) +
  geom_cladelabel(70, "Cercopithecoidea", offset=15, barsize=2, angle=90,
                  offset.text=1.5, hjust=0.5, fontsize=3)
p3
```



There are some helpful details here, such as the fact that the label for Tarsioidea is off at an angle to avoid overlapping with other labels (*angle = 75*). The extra arguments in these options demonstrate how much control you can exercise over each geom.

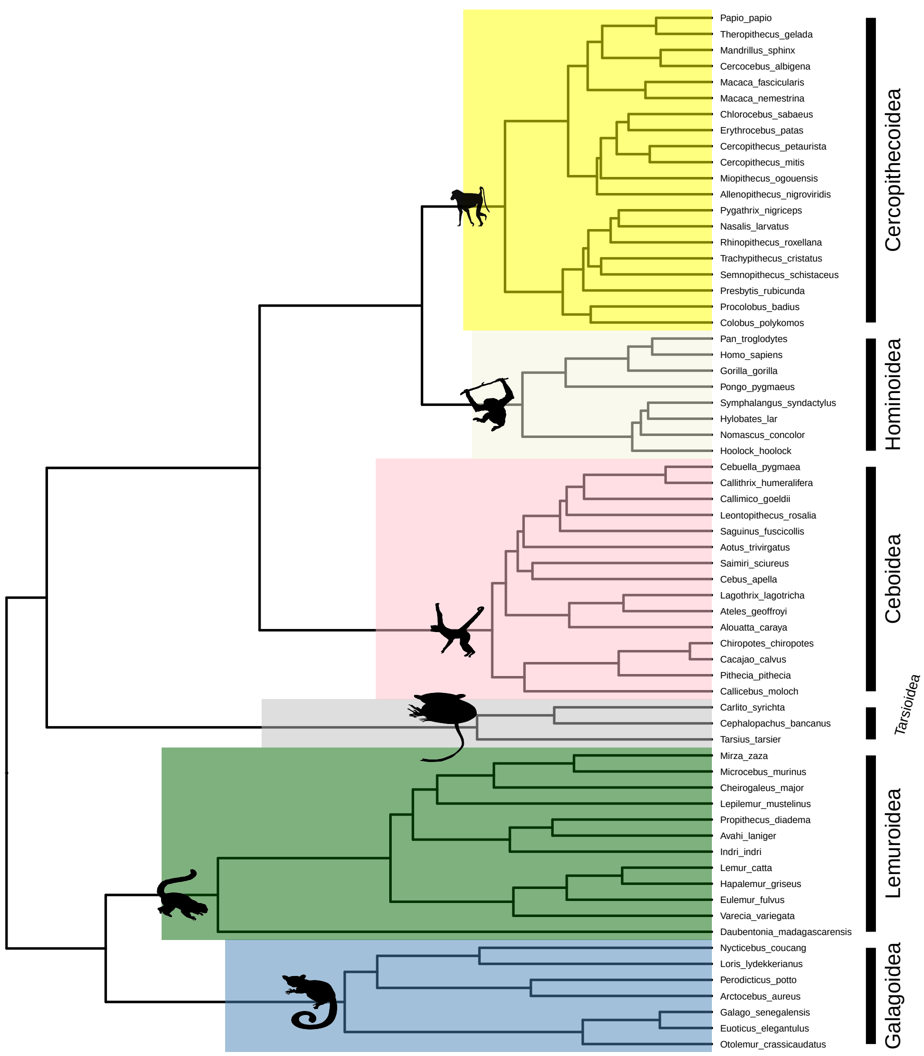
Now let's get to adding images. The way to do this is a little awkward but I think it's worth the hassle. The first thing we have to do is gather the links for each image we want to use. I've chosen to do this by building a small data frame containing the urls to the images on phylopic, the names of the super families I want to label and the nodes I want to plot the images on.

```
images <- data.frame(node = c(124,113,110,96,89,70),
phylopic = c("http://phylopic.org/assets/images/submissions/
7fb9bea8-e758-4986-afb2-95a2c3bf983d.512.png",
"http://phylopic.org/assets/images/submissions/
bac25f49-97a4-4aec-beb6-f542158ebd23.512.png",
"http://phylopic.org/assets/images/submissions/
f598fb39-facf-43ea-a576-1861304b2fe4.512.png",
"http://phylopic.org/assets/images/submissions/
aceb287d-84cf-46f1-868c-4797c4ac54a8.512.png",
"http://phylopic.org/assets/images/submissions/
0174801d-15a6-4668-bfe0-4c421fbe51e8.512.png",
```

```
"http://phylopic.org/assets/images/submissions/
72f2f854-f3cd-4666-887c-35d5c256ab0f.512.png"),
species = c("Galagoidea","Lemuroidea","Tarsioidea",
"Ceboidea","Hominoidea","Cercopithecoidea"))
```

Once we have the urls we need in a nice dataframe, we can pipe them into the **geom_nodelab** geom and the end product should appear.

```
p3 %<+% images +
  geom_nodelab(aes(image = phylopic), geom = "image", size = .04, nudge_x = -4)
```



As you can probably tell, the images don't have to be from Phylopic. You can use any images you have the rights to in exactly the same way!

5.5 Further info

This chapter barely scratches the surface of what ggtree is capable of. For much more detail, have a look at Guangchuang Yu's very own Bookdown covering the topic. You can access the book by clicking here or by running the following code in R once you have ggtree installed.

```r
vignette("ggtree", package = "ggtree")
```

Chapter 6

ANOVA

Analysis of variance (ANOVA) is something you should recognise from your quantitative skills course. This chapter will begin with a brief recap before showing you how to perform phylogenetically corrected ANOVA.

6.1 Analysis of variance

Analysis of variance asks if there are differences in the mean values between 3 or more categories. If there are only two categories (Terrestrial/Aquatic for example), then you need a t-test.

In LIFE223 you analysed the results of an experiment in which corncrake hatchlings were raised on four different supplements in addition to their normal diet.

```
Warning: Ignoring unknown parameters: fun.y
```

```
No summary function supplied, defaulting to `mean_se()`
```

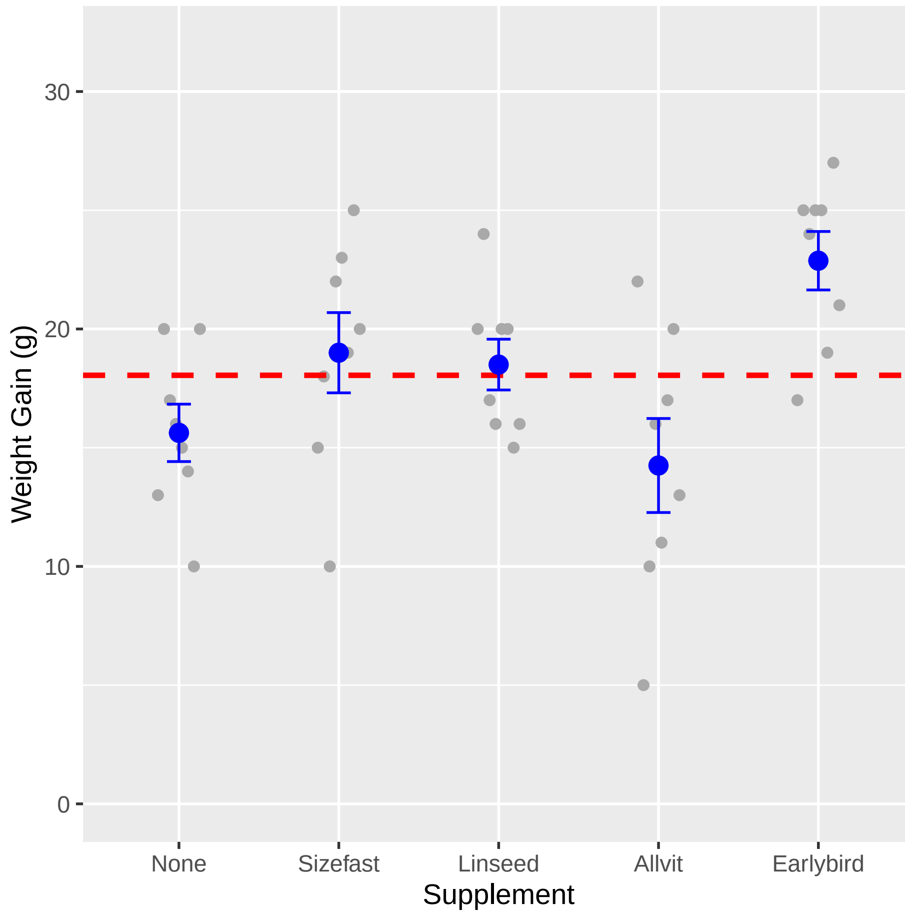


Figure 6.1: Plot of weight gain in corncrake hatchlings reared on four different nutritional supplements and a control group. The mean and standard deviation of each group is plotted in blue. the mean weight gain across the entire sample is plotted in red.

```
corncrake.model <- lm(WeightGain ~ Supplement, data = corncrake)
anova(corncrake.model)
```

```
Analysis of Variance Table

Response: WeightGain
           Df Sum Sq Mean Sq F value   Pr(>F)
Supplement  4 357.65  89.413  5.1281 0.002331 **
Residuals  35 610.25  17.436
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The one-way ANOVA shows that there was a significant effect of supplement on the weight gain of the corncrake hatchlings ($F = 5.1$, $df = 4, 35$, $p < 0.01$). The final step is to perform our multiple comparisons test.

```
corncrake.aov <- aov(corncrake.model)
TukeyHSD(corncrake.aov, ordered = TRUE)
```

```
  Tukey multiple comparisons of means
    95% family-wise confidence level
    factor levels have been ordered

Fit: aov(formula = corncrake.model)

$Supplement
                   diff       lwr       upr     p adj
None-Allvit        1.375 -4.627565  7.377565 0.9638453
Linseed-Allvit     4.250 -1.752565 10.252565 0.2707790
Sizefast-Allvit    4.750 -1.252565 10.752565 0.1771593
Earlybird-Allvit   8.625  2.622435 14.627565 0.0018764
Linseed-None       2.875 -3.127565  8.877565 0.6459410
Sizefast-None      3.375 -2.627565  9.377565 0.4971994
Earlybird-None     7.250  1.247435 13.252565 0.0113786
Sizefast-Linseed   0.500 -5.502565  6.502565 0.9992352
Earlybird-Linseed  4.375 -1.627565 10.377565 0.2447264
Earlybird-Sizefast 3.875 -2.127565  9.877565 0.3592201
```

The **TukeyHSD** functions shows us the pairwise comparisons between groups. We can see (for example) that *Allvit* was not significantly different from the control (difference = 1.375g, $p = 0.96$) but *Earlybird* was significantly better than the control group (difference = 7.25g, $p = 0.01$).

6.2 Phylogenetic correction

As you know, when trying to run a similar analysis on non-independent data (such as species) we will run into problems. Garland *et al* [1993] developed a simulation based approach to solve this problem. The phylogenetic ANOVA uses computer simulations of traits evolving the phylogenetic tree. The next section contains some example data and a phylogeny to demonstrate the method.

6.3 Example data & analysis

The data we're using is taken from the package geiger [Harmon et al., 2008] so make sure the package is installed and loaded.

```r
install.packages("geiger")
library(geiger)
```

Load the data as follows. The tree and data are stored together so we'll need to save the to separate objects called **dat** and **phy**. You probably don't *need* to do this but this is more similar to what you're likely to see when using your own data.

```r
data("geospiza")
dat <- geospiza$dat
tree <- geospiza$phy
head(dat)
```

```
                wingL  tarsusL  culmenL    beakD   gonysW
magnirostris 4.404200 3.038950 2.724667 2.823767 2.675983
conirostris  4.349867 2.984200 2.654400 2.513800 2.360167
difficilis   4.224067 2.898917 2.277183 2.011100 1.929983
scandens     4.261222 2.929033 2.621789 2.144700 2.036944
fortis       4.244008 2.894717 2.407025 2.362658 2.221867
fuliginosa   4.132957 2.806514 2.094971 1.941157 1.845379
```

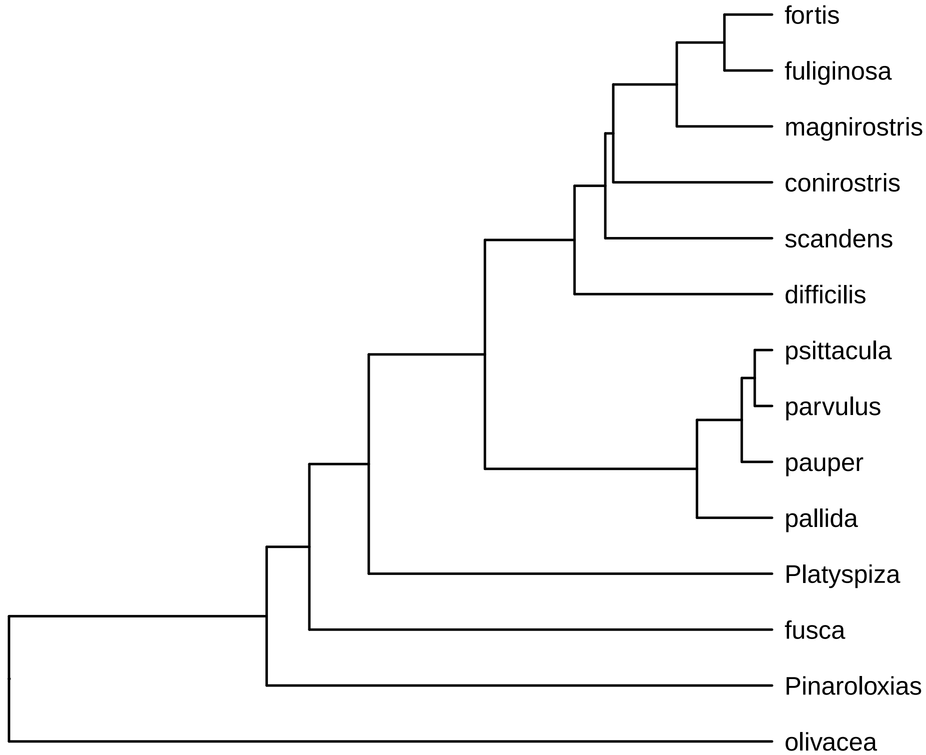


Figure 6.2: Phylogeny of the species contained within the 'geospiza' dataset of the package geiger.

We need to start by defining the categories for the data. It is likely that you will have already done this in your data frame. If so, just make sure the groups are stored as a factor. In this case, we'll just create some random categories to work with for the example.

```
groups <- as.factor(c(rep("A", 4), rep("B", 5), rep("C", 4)))
names(groups) <- rownames(dat)
```

An important step here (and for every phylogenetic analysis) is making sure the tree and data can be compared. To do this, we should make sure that the rownames of the data are species names and not just numbers. In this case they already are but if they aren't for you data, you can use the following code.

```
rownames(data) <- data$SPECIES #the column with species names in the data
```

The geiger package has a very useful function called **name.check** to allow us to check that the rownames of our data match the tip labels of our tree.

```
name.check(tree, dat)
```

```
$tree_not_data
[1] "olivacea"

$data_not_tree
character(0)
```

We can see that *olivacea* is not in our data. For some analyses, mismatches like this are a problem and you will need to drop the tip from the tree. It actually doesn't matter here because the function we will be using can drop it automatically for us. However, let's see how it's done. Note the use of the function **drop.tip** from the package **ape** [Paradis and Schliep, 2018] which is an essential package to have for this kind of work!

```
tree <- ape::drop.tip(tree, tip = "olivacea")
```

Now we have overwritten the old tree with our pruned tree. Let's check the new one matches the data.

```
name.check(tree, dat)
```

```
[1] "OK"
```

All that's left now is to run the analysis. First we extract the column of interest from our data and then simply use the function **aov.phylo**.

```
d1 <- dat[,1]
```

You should notice some similarities and differences from the way you have run ANOVA before. We are still using a formula (the part with $\sim$) but not in a separate **lm** function. We need to specify the tree we want to use (**tree**) and also how many simulations we want to run. There isn't a firm rule about this but general convention is around 1000 when sampling/bootstrapping/simulations are involved.

```
x <- aov.phylo(d1 ~ groups, phy = tree, nsim = 1000)
```

```
Analysis of Variance Table

Response: dat
          Df   Sum-Sq  Mean-Sq F-value  Pr(>F) Pr(>F) given phy
group      2 0.063237 0.031619  3.0067 0.09497           0.1538
Residuals 10 0.105161 0.010516
```

The results table should be very familiar! The only real difference here is that you have been provided with two p-values. The first (*Pr(>F)*) is the p-value without accounting for phylogeny and the second (*Pr(>F) given phy*) is the value when we account for phylogeny. In both cases, there is no significant difference between groups.

As you can see, accounting for phylogeny *usually* raises the p-value (makes it less significant). This shows us that not accounting for phylogeny increases the risk of type I errors (false positives).

6.4 Further info

For further information about the phylogenetic ANOVA, you can read the original paper by Garland *et al* [1993].

Chapter 7

Ancestral State Reconstruction I

This chapter will take you through the code we can use to run ancestral state reconstruction with **categorical** characters. As always, remember to begin by setting your working directory to wherever you have saved the data files.

7.1 Data

The first thing we need to do is load some data. When you're doing this, you need to keep in mind that you should keep your workspace as well organised as possible. In practice, this means giving things good names. "RicksDataV1.1" is not a great name depending on how many datasets you want in there. Neither is "data1" if you plan on having multiple datasets (which we do). So give your data object, and all other objects, simple, useful names. My personal preference is to use the name of the group but whatever works is fine. You need to be able to keep track of everything.

```
macaques <- read.table("macaque_data.txt", header = TRUE)
```

In your environment panel there should be a data frame with 16 observations of 2 variables. This command will show us the top 6 rows of data. It's helpful to have a quick look and see R has loaded what we expected. In this case our data contains 15 species of macaque and one species of baboon alongside data regarding whether they exhibit sexual swellings or not (1/0).

```
head(macaques)
```

```
             species swelling
1    Macaca_arctoides        0
```

```
2   Macaca_assamensis        0
3     Macaca_cyclopis        1
4 Macaca_fascicularis        1
5      Macaca_fuscata        1
6        Macaca_maura        1
```

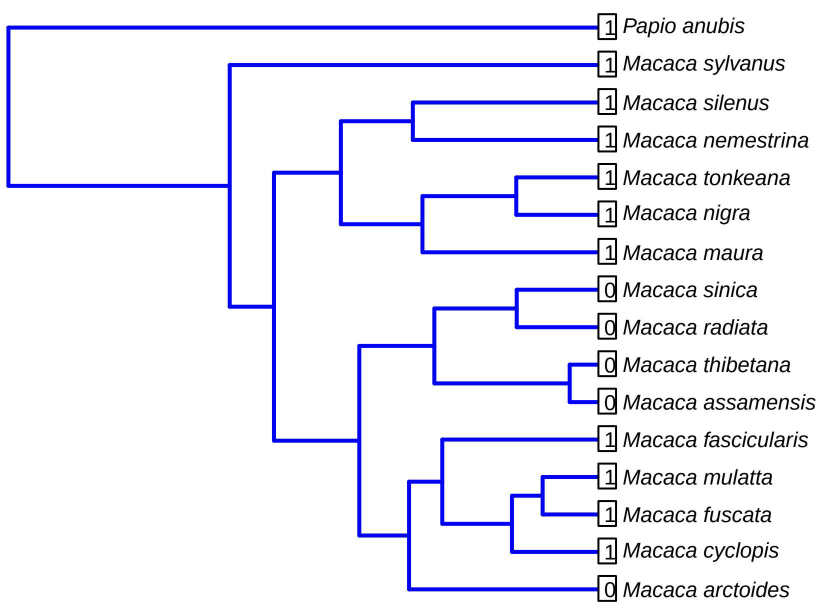
7.2 Trees

Now we need to load the tree using the **read.nexus** function in the package **ape** [Paradis and Schliep, 2018].

```
macaque.tree <- read.nexus("macaque_tree.nex")
```

Let's plot the tree to make sure it loaded correctly. I've used base graphics here rather than ggtree (annotated to let you know what it does). Feel free to have a mess around with these options so you get a feel for what they do. The second function "tiplabels" adds some extra tip labels containing the data from the second column of our macaque data.

```
plot(macaque.tree,          #Tree object
     cex = 0.7,             #Font size for tip labels
     label.offset = 0.3,    #Create a space between tip and label
     edge.color = "blue",   #Paint the branches blue
     edge.width = 2)        #Make the branches thicker
tiplabels(macaques[,2], bg = "white", cex = 0.7)
```



7.3 Parsimony

Let's first generate the most parsimonious reconstruction of the history of this trait. Remember that the most parsimonious history is the one that has the fewest evolutionary transitions. Parsimony is conceptually based upon Occam's razor which states that all else being equal, the simplest explanantion is always the correct one.

The function for this is **MPR**. It takes an unrooted tree and asks you to specify the root. In our case, we'll have to unroot our tree and then re-root it by specifying that *Papio anubis* is our outgroup.

```
mp1 <- MPR(macaques[,2], unroot(macaque.tree), "Papio_anubis")
```

When we investigate mp1, we can see a list of results matched up to numbered nodes on the tree. Some nodes are clearly in state 1 and others in state 0. Interestingly some are indeterminate and could be either 0 or 1 such as nodes 19 and 20.

```
mp1
```

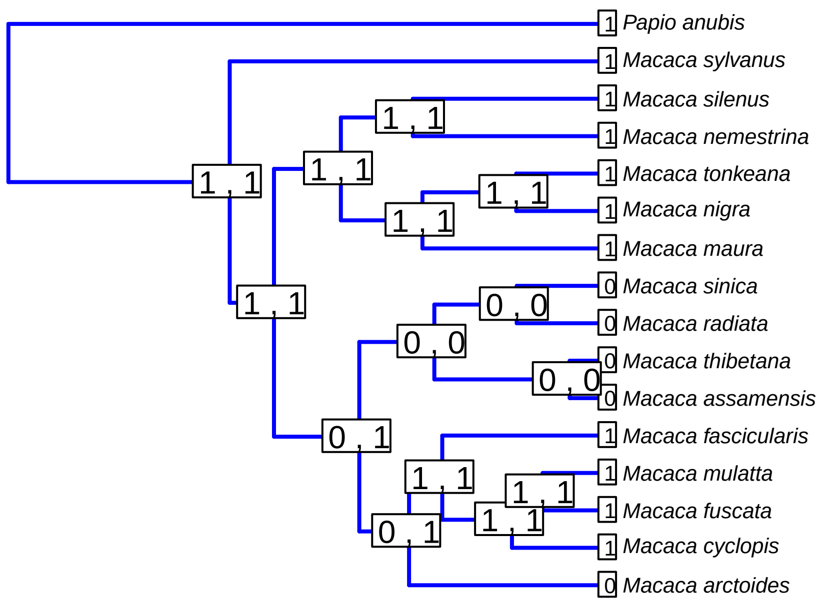
```
   lower upper
17     1     1
18     1     1
19     0     1
20     0     1
21     1     1
22     1     1
23     1     1
24     0     0
25     0     0
26     0     0
27     1     1
28     1     1
29     1     1
30     1     1
```

To get an idea of what this means, we should plot it on the tree. This loop cycles through our results list and combines the lower and upper estimates for each node into a text string that we can then overlay onto that node.

```
mp.nodes <- numeric(0)
for(i in 1:length(mp1[,1])){
  mp.nodes <- append(mp.nodes, paste(mp1[i,1], ",", mp1[i,2]))
}
```

Once we've done that we can plot those expressions onto the tree with the function nodelabels.

```
plot(macaque.tree, cex = 0.7, label.offset = 0.3,
     edge.color = "blue", edge.width = 2)
tiplabels(macaques[,2], bg = "white", cex = 0.7)
nodelabels(mp.nodes, c(18:31), bg = "white")
```



You should note that this isn't a very good plot! There are better ways to represent this information with a little code manipulation. Here's a version using **ggtree** that plots the character states as points on the tips and the reconstructed nodes.

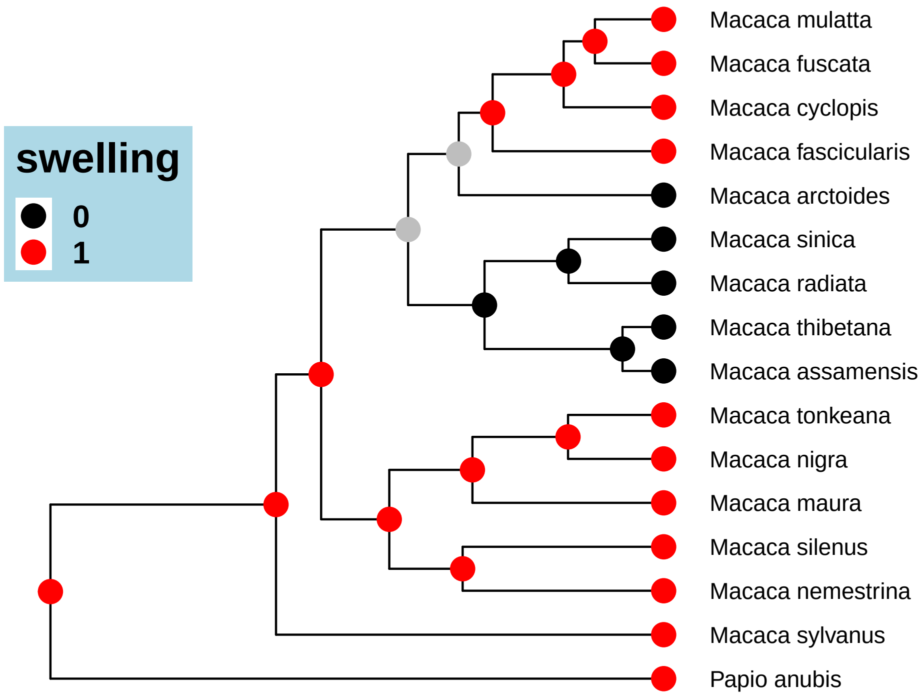


Figure 7.1: Maximum parsimony reconstruction of the evolution of conspicuous sexual swellings in macaques

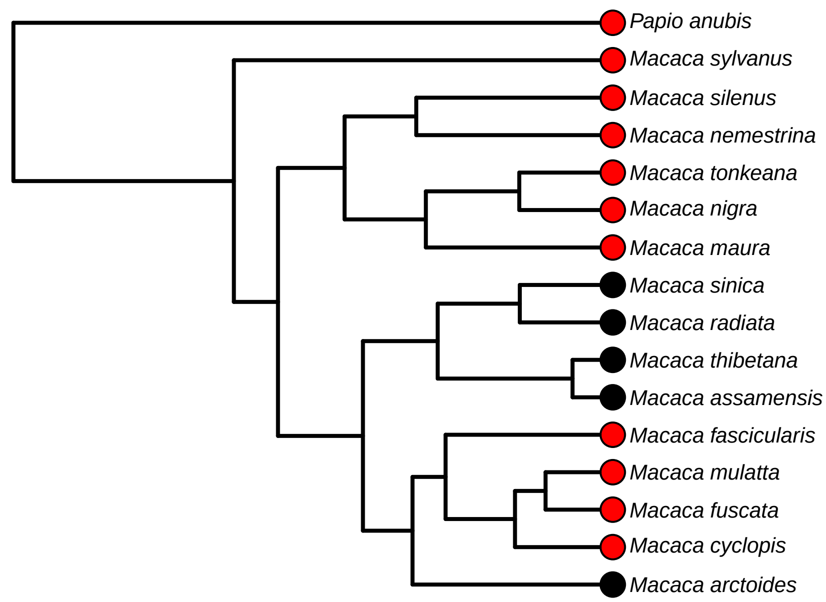
As you can see, the uncertainty in some nodes comes from the fact that there seems to be at least two equally parsimonious histories with gains and losses ocurring in different places. For any serious analysis, this is a highly unsatisfactory outcome!

7.4 Maximum Likelihood

Let's try a different approach. Maximum likelihood is different from parsimony for many reasons but most significantly, it can make use of branch length information. This is very useful in discriminating between possible histories. A longer branch means more evolutionary change (either in time or character change) and so transitions are more likely to occur on longer branches.

Let's replot the tree. Here I've changed the tiplabels function to plot the character states as colours rather than numbers. The **bg** argument is what lets me do this. In this argument I list the states (adding 1 because the first is 0) and then the function passes those states to R to assign colours based on a numbered list of standard colours.

```
plot(macaque.tree, cex = 0.7, label.offset = 0.4, edge.width = 2)
tiplabels(pch = 21, bg = as.numeric(macaques$swelling)+1, cex = 1.7)
```



To run an ancestral state reconstruction using maximum likelihood we can use the function **ace** (ancestral character estimation) in the ape package [Paradis and Schliep, 2018]. In our first reconstruction, we will make the assumption that the rate of evolution of the trait is equal across the tree by setting the model to *ER* (equal rates).

```
m1 <- ace(x = macaques$swelling,  #trait data
          phy = macaque.tree,     #phylogeny
          method = "ML",          #method (Maximum likelihood)
          type = "discrete",      #type of data (continuous or discrete)
          model = "ER")           #Model of evolution
m1
```

```

    Ancestral Character Estimation

Call: ace(x = macaques$swelling, phy = macaque.tree, type = "discrete",
    method = "ML", model = "ER")

    Log-likelihood: -6.906593

Rate index matrix:
  0 1
0 . 1
1 1 .
```

```
Parameter estimates:
 rate index estimate std-err
          1   0.0319  0.0191

Scaled likelihoods at the root (type '...$lik.anc' to get them for all nodes):
         0          1
0.08625654 0.91374346
```

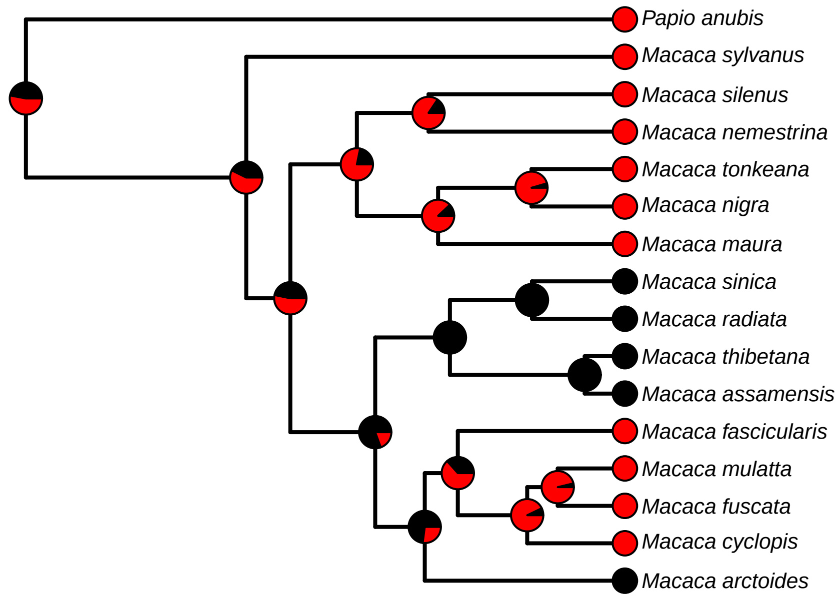
Looking at the results shows us the likelihood at the root (91% in favour of state 1 here). However, it's always best to plot the results. We can represent the likelihoods at each node with a piechart. Generally speaking, piecharts are awful but when used in this way, they can actually add useful information to a plot and that's the most important point about plotting any data. In this plot, the piecharts represent the probability that each node exhibited sexual swelling (red) or concealed estrus (black). We can see that the two uncertain nodes from our parsimony analysis are now more certain. Visual inspection shows that these nodes have a greater tha 75% probability of having exhibited sexual swellings.

```r
plot(macaque.tree, cex = 0.7, label.offset = 0.4, edge.width = 2)
tiplabels(pch = 21, bg = as.numeric(macaques$swelling)+1, cex = 1.7)
nodelabels(pie = m1$lik.anc, piecol = c("black", "red"), cex = 0.8)
```

Now we can run a similar analysis but let's assume that rates of evolution can vary by setting model to **ARD** (All Rates Different).

```r
m2 <- ace(x = macaques$swelling, phy = macaque.tree,
          method = "ML", type = "discrete", model = "ARD")

plot(macaque.tree, cex = 0.7, label.offset = 0.4, edge.width = 2)
tiplabels(pch = 21, bg = as.numeric(macaques$swelling)+1, cex = 1.7)
nodelabels(pie = m2$lik.anc, piecol = c("black", "red"), cex = 0.8)
```



As you can see, the different model of evolution makes a big difference to the results. Which model you choose to use depends on which assumptions you think are justified. Is it fair to assume that the rate of evolution of conspicuous sexual swelling would be constant across the tree as in the equal rates model?

7.5 Stochastic Character Mapping

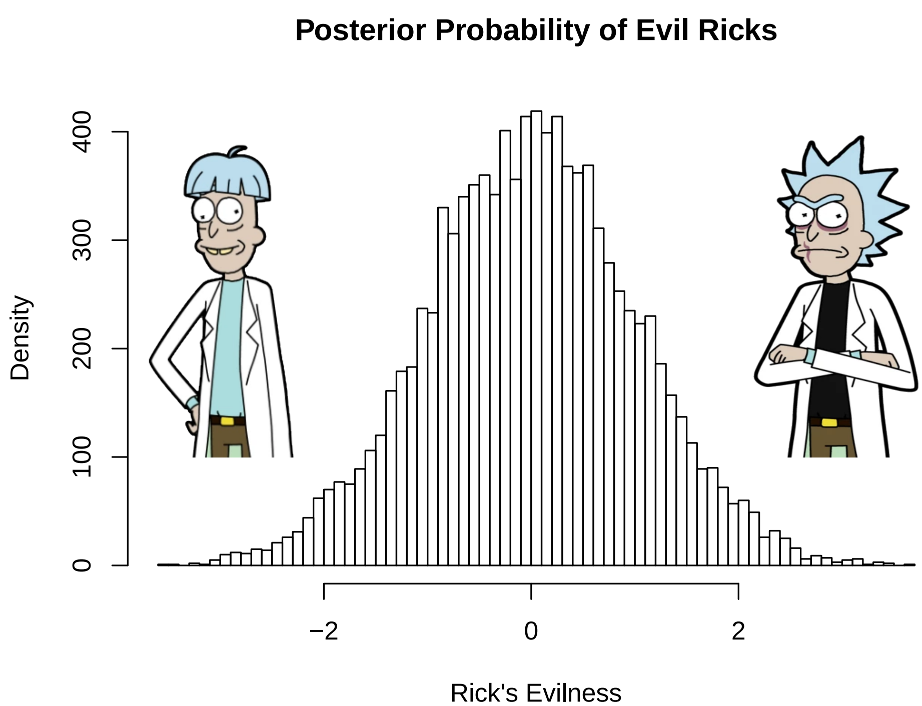
Stochastic character mapping uses an **MCMC** (Markov chain Monte-Carlo) approach to sample possible reconstructions from a posterior probability distribution.

Think of the posterior probability distribution as containing all the possible evolutionary histories of the trait in question. This includes some histories in which everything was in one state right up until a few generations from the present when everything swapped around at the same time to give us the distribution we see today. It also contains a history in which the trait switches between 0 and 1 every other generation essentially at random.

Obviously these kind of histories are biologically absurd but not mathematically impossible. They have low statistical probability. Certain other histories will have a high statistical probability and so there will be many similar histories in the distribution. The distribution can be thought of as a histogram with some parameter that defines each particular history.

7.5.1 An Analogy

Let's say that we were to plot the entire multiverse as such a distribution using the evil tendencies of one particular occupant (Rick Sanchez) of the multiverse as our parameter. All the different Ricks in all the different universes will vary in their evil tendencies. But overall, Rick's character is actually a nihilist meaning his mean evilness is around 0 when taken over the whole multiverse. Given all this, the posterior distribution of evil Ricks in the multiverse might look like this.



MCMC samples this distribution of histories in a chain. If a history has a higher likelihood than the previous sampling, it is accepted. If it is lower then it is rejected from the sample. In this way, MCMC quickly narrows down the possibilities and gives us a sample of quite likely histories.

7.5.2 2-State Characters

Let's see it in action. We'll need the **phytools** package [Revell, 2012] to create our stochastic character map.

```
library(phytools)
```

For this analysis (like other phytools functions) we'll need our data in a named vector rather than a data table. Let's call it swelling. The **names** function attaches the species name to each value in our new vector.

```
swelling <- macaques$swelling
names(swelling) <- macaques$species
swelling
```

```
   Macaca_arctoides   Macaca_assamensis     Macaca_cyclopis Macaca_fascicularis
                  0                   0                   1                   1
     Macaca_fuscata        Macaca_maura      Macaca_mulatta   Macaca_nemestrina
                  1                   1                   1                   1
       Macaca_nigra      Macaca_radiata      Macaca_silenus       Macaca_sinica
                  1                   0                   1                   0
    Macaca_sylvanus    Macaca_thibetana     Macaca_tonkeana         Papio_anubis
                  1                   0                   1                   1
```

Now we can sample character histories assuming an *equal rates* model of evolution using the **make.simmap** function.

```
scm1 <- make.simmap(macaque.tree, x = swelling, model = "ER")
```

```
make.simmap is sampling character histories conditioned on the transition matrix

Q =
            0           1
0 -0.03185011  0.03185011
1  0.03185011 -0.03185011
(estimated using likelihood);
and (mean) root node prior probabilities
pi =
  0   1
0.5 0.5


Done.
```

Q here is the matrix of transition rates which we have constrained to be equal (model = "ER") which explains why the numbers match. As usual with reconstructions, the best thing is to plot them. Here we can use the phytools function **plotSimmap** to plot the special object we've created. It even has a companion function to add a legend. The first line here assigns colours to the traits.

```
cols <- setNames(c("black", "red"), sort(unique(swelling)))
plotSimmap(scm1, cols, pts = F, lwd = 3, fsize = .8)
add.simmap.legend(colors = cols, vertical = F, prompt = F, x = 0, y = 10, fsize = .8)
```

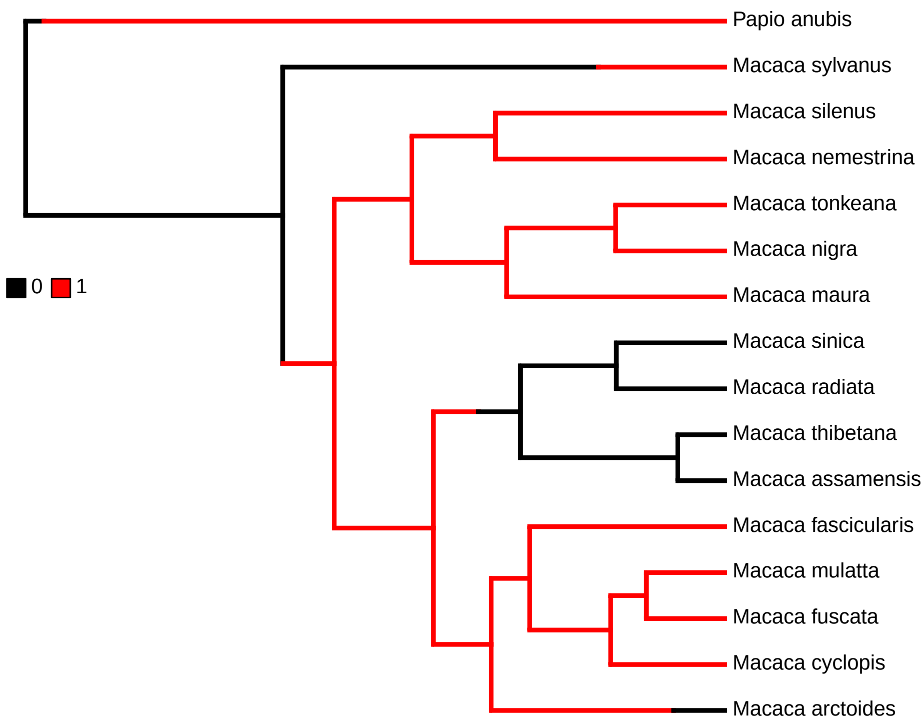


Figure 7.2: Simmap showing a single possible evolutionary history of sexual swelling in macaques.

Here you can see the single history we have sampled (yours will likely differ). The history contains branches painted according to the trait colour we specified and the position of the transitions on the branch mark the exact position the changes are theorised to have taken place. This is an awful lot of certainty for an ancestral state reconstruction! You should note that the one plotted here is very odd. It says that the ancestor of the group had concealed estrus and then this trait was lost 3 times independently, leaving no trace in the extant species. Given the data and tree we provided, it is hard to see how we can have any confidence in this reconstruction. What evidence have we collected that actually supports this?

However, we need to remember that this only one of the many possible histories! Our next step should be to extract a reasonable sample of these histories!

Let's sample 500 and when R has done that, we can use **describe.simmap** to summarize the sample.

```
scm2 <- make.simmap(macaque.tree, swelling, model = "ER", nsim = 500)
```

```
make.simmap is sampling character histories conditioned on the transition matrix
```

```
Q =
            0           1
0 -0.03185011  0.03185011
1  0.03185011 -0.03185011
(estimated using likelihood);
and (mean) root node prior probabilities
pi =
  0   1
0.5 0.5

Done.
```

```
scm2.sum <- describe.simmap(scm2, plot = FALSE)
```

When we call up the summary, we can see some interesting details about our sample. It seems to be saying that transitions from 1 to 0 (a loss of sexual swelling) happen more frequently than gains of sexual swelling.

```
scm2.sum
```

```

500 trees with a mapped discrete character with states:
 0, 1

trees have 2.872 changes between states on average

changes are of the following types:
       0,1   1,0
x->y 0.758 2.114

mean total time spent in each state is:
              0          1   total
raw  17.6835904 71.4988096 89.1824
prop  0.1982857  0.8017143  1.0000
```

As usual, we're going to want a summary plot. The backbone of this plot won't look quite the same as the previous one. You don't want confusing information on your plot so here it would be better to plot a blank backbone (ie a tree with just one colour of branch that doesn't match the colour of the traits) and represent the trait transitions as we did previously with pie charts. In this case the pies represent the proportion of histories in each state (1 or 0) at each node.

```
cols.null <- setNames(c("darkgrey", "darkgrey"), sort(unique(swelling)))
plotSimmap(scm2[[1]], lwd = 3, pts = F, setEnv = T, colors = cols.null, offset = .6)
nodelabels(pie = scm2.sum$ace, piecol = cols, cex = 0.6)
add.simmap.legend(colors = cols, vertical = F, prompt = F, x = 0, y = 10, fsize = .8)
tiplabels(pch = 21, bg = as.numeric(macaques$swelling)+1, cex = 2)
```

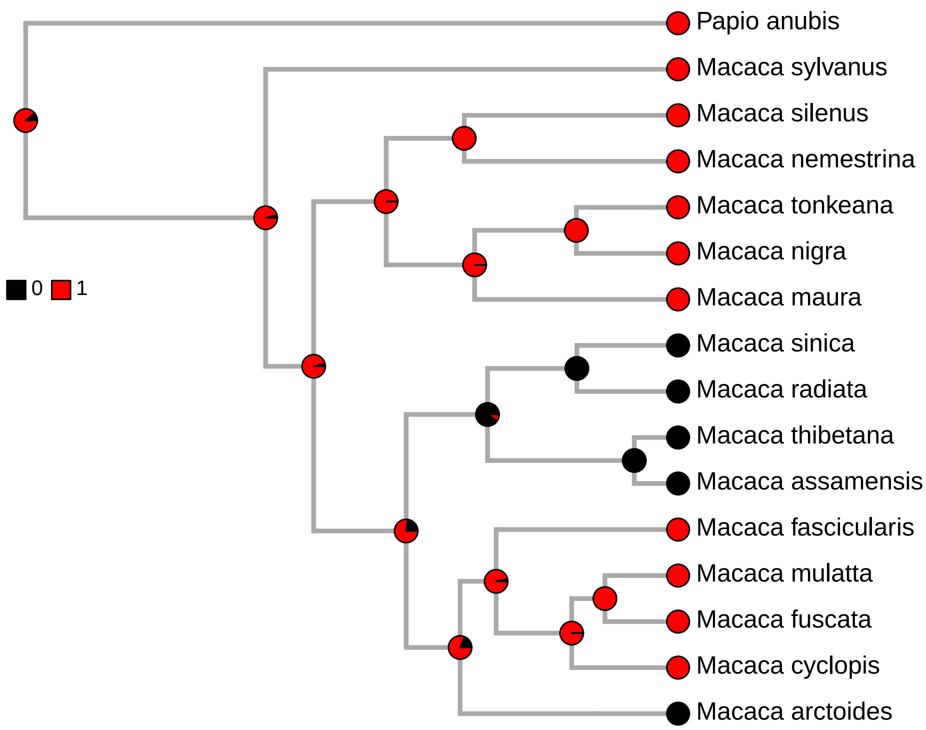


Figure 7.3: Summary of 500 sampled discrete character histories showing the evolution of sexual swellings in macaques.

This analysis gives us a very similar output to the maximum likelihood analysis in the previous section. If you're intrested, give this analysis another try with different models of evolution.

7.5.3 3-State Characters

Stochastic character mapping can also be used for traits with more than one state. For example, burrowing in carnivores can be classified as 0 (no burrowing), 1 (use a burrow dug by another animal) or 2 (dig your own burrow).

7.5.3.1 Data

Let's load some data from a paper which investigated aposematism in terrestrial carnivores [Stankowich et al., 2011]. Don't forget to assign the species names to rownames to keep everything tidy while we manipulate the data. We also have a tree covering all carnivores [Nyakatura and Bininda-Emonds, 2012].

```
carn.tree <- read.nexus("carnivores_tree.nex")
carn.data <- read.table("carnivores_data.txt", header = T)
rownames(carn.data) <- carn.data$Species
```

If you look at the new object **carn.tree** you'll notice it is a multiPhylo object. This means it actually contains a number of trees rather than just one. For more details about this class of object, see chapter 3.

For now, we just want the first one in the list (based on the best estimates used to date the tree). I'll also prune it a bit to get rid of some of the species I'm not interested in for now.

```
carn.tree <- carn.tree[[1]]
carn.tree <- extract.clade(carn.tree, node = "'123'")
```

Unlike the macaque data from earlier, the carnivore data needs a little more tidying. Now that you're more comfortable using R, you should make this standard practice whenever you load data and a tree for an analysis!

We can use the function **name.check** in the package **geiger** to help us out here [Harmon et al., 2008]. This function returns two lists. The first contains all the species that appear in the phylogeny but not in the dataset. The second has the species that occur in the data but not in the tree.

```
geiger::name.check(phy = carn.tree, data = carn.data)
```

```
$tree_not_data
 [1] "Arctocephalus_australis"     "Arctocephalus_forsteri"
 [3] "Arctocephalus_galapagoensis" "Arctocephalus_gazella"
 [5] "Arctocephalus_philippii"     "Arctocephalus_pusillus"
 [7] "Arctocephalus_townsendi"     "Arctocephalus_tropicalis"
 [9] "Bassaricyon_alleni"          "Bassaricyon_beddardi"
[11] "Bassaricyon_lasius"          "Bassaricyon_pauli"
[13] "Callorhinus_ursinus"         "Conepatus_chinga"
[15] "Conepatus_humboldtii"        "Conepatus_semistriatus"
[17] "Cystophora_cristata"         "Dusicyon_australis"
[19] "Erignathus_barbatus"         "Eumetopias_jubatus"
[21] "Halichoerus_grypus"          "Histriophoca_fasciata"
[23] "Hydrurga_leptonyx"           "Leptonychotes_weddellii"
[25] "Lobodon_carcinophaga"        "Lontra_provocax"
[27] "Lutra_nippon"                "Lutra_sumatrana"
[29] "Lycalopex_fulvipes"          "Lycalopex_griseus"
[31] "Lycalopex_gymnocercus"       "Lycalopex_sechurae"
[33] "Lyncodon_patagonicus"        "Martes_gwatkinsii"
[35] "Meles_anakuma"               "Meles_leucurus"
[37] "Melogale_everetti"           "Melogale_orientalis"
[39] "Melogale_personata"          "Mirounga_angustirostris"
[41] "Mirounga_leonina"            "Monachus_monachus"
```

```
[43] "Monachus_schauinslandi"      "Monachus_tropicalis"
[45] "Mustela_felipei"             "Mustela_itatsi"
[47] "Mustela_kathiah"             "Mustela_lutreolina"
[49] "Mustela_nudipes"             "Mustela_strigidorsa"
[51] "Mustela_subpalmata"          "Nasuella_olivacea"
[53] "Neophoca_cinerea"            "Neovison_macrodon"
[55] "Odobenus_rosmarus"           "Ommatophoca_rossii"
[57] "Otaria_flavescens"           "Pagophilus_groenlandicus"
[59] "Phoca_largha"                "Phoca_vitulina"
[61] "Phocarctos_hookeri"          "Procyon_pygmaeus"
[63] "Pusa_caspica"                "Pusa_hispida"
[65] "Pusa_sibirica"               "Spilogale_angustifrons"
[67] "Urocyon_littoralis"          "Vulpes_bengalensis"
[69] "Vulpes_ferrilata"            "Zalophus_californianus"
[71] "Zalophus_japonicus"          "Zalophus_wollebaeki"

$data_not_tree
 [1] "Acinonyx_jubatus"             "Arctictis_binturong"
 [3] "Arctogalidia_trivirgata"      "Atilax_paludinosus"
 [5] "Bdeogale_crassicauda"         "Caracal_caracal"
 [7] "Catopuma_temminckii"          "Chrotogale_owstoni"
 [9] "Civettictis_civetta"          "Crocuta_crocuta"
[11] "Crossarchus_obscurus"         "Cryptoprocta_ferox"
[13] "Cynictis_penicillata"         "Cynogale_bennettii"
[15] "Dologale_dybowskii"           "Eupleres_goudotii"
[17] "Felis_chaus"                  "Felis_manul"
[19] "Felis_margarita"              "Felis_nigripes"
[21] "Felis_silvestris"             "Fossa_fossana"
[23] "Galerella_sanguinea"          "Galidia_elegans"
[25] "Genetta_abyssinica"           "Genetta_angolensis"
[27] "Genetta_genetta"              "Genetta_servalina"
[29] "Genetta_thierryi"             "Helogale_parvula"
[31] "Hemigalus_derbyanus"          "Herpestes_ichneumon"
[33] "Herpestes_javanicus"          "Herpestes_urva"
[35] "Hyaena_brunnea"               "Hyaena_hyaena"
[37] "Ichneumia_albicauda"          "Leopardus_geoffroyi"
[39] "Leopardus_guigna"             "Leopardus_jacobitus"
[41] "Leopardus_pardalis"           "Leopardus_wiedii"
[43] "Leptailurus_serval"           "Liberiictis_kuhni"
[45] "Lynx_canadensis"              "Lynx_lynx"
[47] "Lynx_pardinus"                "Lynx_rufus"
[49] "Macrogalidia_musschenbroekii" "Mungos_gambianus"
[51] "Mungos_mungo"                 "Mungotictis_decemlineata"
[53] "Nandinia_binotata"            "Neofelis_nebulosa"
[55] "Paguma_larvata"               "Panthera_leo"
[57] "Panthera_onca"                "Panthera_pardus"
```

```
[59] "Panthera_tigris"               "Paracynictis_selousi"
[61] "Paradoxurus_hermaphroditus"    "Paradoxurus_zeylonensis"
[63] "Pardofelis_marmorata"          "Poiana_richardsonii"
[65] "Prionailurus_bengalensis"      "Prionailurus_iriomotensis"
[67] "Prionailurus_rubiginosus"      "Prionodon_linsang"
[69] "Prionodon_pardicolor"          "Proteles_cristata"
[71] "Puma_concolor"                 "Salanoia_concolor"
[73] "Suricata_suricatta"            "Uncia_uncia"
[75] "Viverra_megaspila"             "Viverra_tangalunga"
[77] "Viverra_zibetha"               "Viverricula_indica"
```

The easiest thing to do first is drop the tips from the tree that we're not interested in. We can pass the whole list to the **drop.tip** function in **ape** for this [Paradis and Schliep, 2018].

```
carn.tree <- drop.tip(carn.tree, geiger::name.check(carn.tree, carn.data)$tree_not_data
geiger::name.check(carn.tree, carn.data)
```

```
$tree_not_data
character(0)

$data_not_tree
 [1] "Acinonyx_jubatus"              "Arctictis_binturong"
 [3] "Arctogalidia_trivirgata"       "Atilax_paludinosus"
 [5] "Bdeogale_crassicauda"          "Caracal_caracal"
 [7] "Catopuma_temminckii"           "Chrotogale_owstoni"
 [9] "Civettictis_civetta"           "Crocuta_crocuta"
[11] "Crossarchus_obscurus"          "Cryptoprocta_ferox"
[13] "Cynictis_penicillata"          "Cynogale_bennettii"
[15] "Dologale_dybowskii"            "Eupleres_goudotii"
[17] "Felis_chaus"                   "Felis_manul"
[19] "Felis_margarita"               "Felis_nigripes"
[21] "Felis_silvestris"              "Fossa_fossana"
[23] "Galerella_sanguinea"           "Galidia_elegans"
[25] "Genetta_abyssinica"            "Genetta_angolensis"
[27] "Genetta_genetta"               "Genetta_servalina"
[29] "Genetta_thierryi"              "Helogale_parvula"
[31] "Hemigalus_derbyanus"           "Herpestes_ichneumon"
[33] "Herpestes_javanicus"           "Herpestes_urva"
[35] "Hyaena_brunnea"                "Hyaena_hyaena"
[37] "Ichneumia_albicauda"           "Leopardus_geoffroyi"
[39] "Leopardus_guigna"              "Leopardus_jacobitus"
[41] "Leopardus_pardalis"            "Leopardus_wiedii"
[43] "Leptailurus_serval"            "Liberiictis_kuhni"
[45] "Lynx_canadensis"               "Lynx_lynx"
[47] "Lynx_pardinus"                 "Lynx_rufus"
[49] "Macrogalidia_musschenbroekii" "Mungos_gambianus"
```

```
[51] "Mungos_mungo"                    "Mungotictis_decemlineata"
[53] "Nandinia_binotata"               "Neofelis_nebulosa"
[55] "Paguma_larvata"                  "Panthera_leo"
[57] "Panthera_onca"                   "Panthera_pardus"
[59] "Panthera_tigris"                 "Paracynictis_selousi"
[61] "Paradoxurus_hermaphroditus"      "Paradoxurus_zeylonensis"
[63] "Pardofelis_marmorata"            "Poiana_richardsonii"
[65] "Prionailurus_bengalensis"        "Prionailurus_iriomotensis"
[67] "Prionailurus_rubiginosus"        "Prionodon_linsang"
[69] "Prionodon_pardicolor"            "Proteles_cristata"
[71] "Puma_concolor"                   "Salanoia_concolor"
[73] "Suricata_suricatta"              "Uncia_uncia"
[75] "Viverra_megaspila"               "Viverra_tangalunga"
[77] "Viverra_zibetha"                 "Viverricula_indica"
```

Dropping species from your dataframe is a little more complex (and in truth not always necessary). One way of doing this is to create a **for loop** that will cycle through the list above and take a subset of the dataframe each time, removing the species in the list as it goes. There are better ways to do this but it might be helpful to become familiar with for loops which are a useful programming tool!

```
pruned.data <- carn.data
for(i in 1:length(geiger::name.check(carn.tree, carn.data)$data_not_tree)){
  pruned.data <- subset(pruned.data, Species!=geiger::name.check(carn.tree, carn.data)$data_not_t
}
geiger::name.check(carn.tree, pruned.data)
```

```
[1] "OK"
```

Once your tree and data are cleaned up we're ready to go!

7.5.3.2 Analysis

As before we need to create a named vector for analysis.

```
burrow<-pruned.data$Burrowing
names(burrow)<-pruned.data$Species
```

Now we can sample a single history and plot it, this time with three colours!

```
scm3<-make.simmap(carn.tree, burrow, model="ER")
```

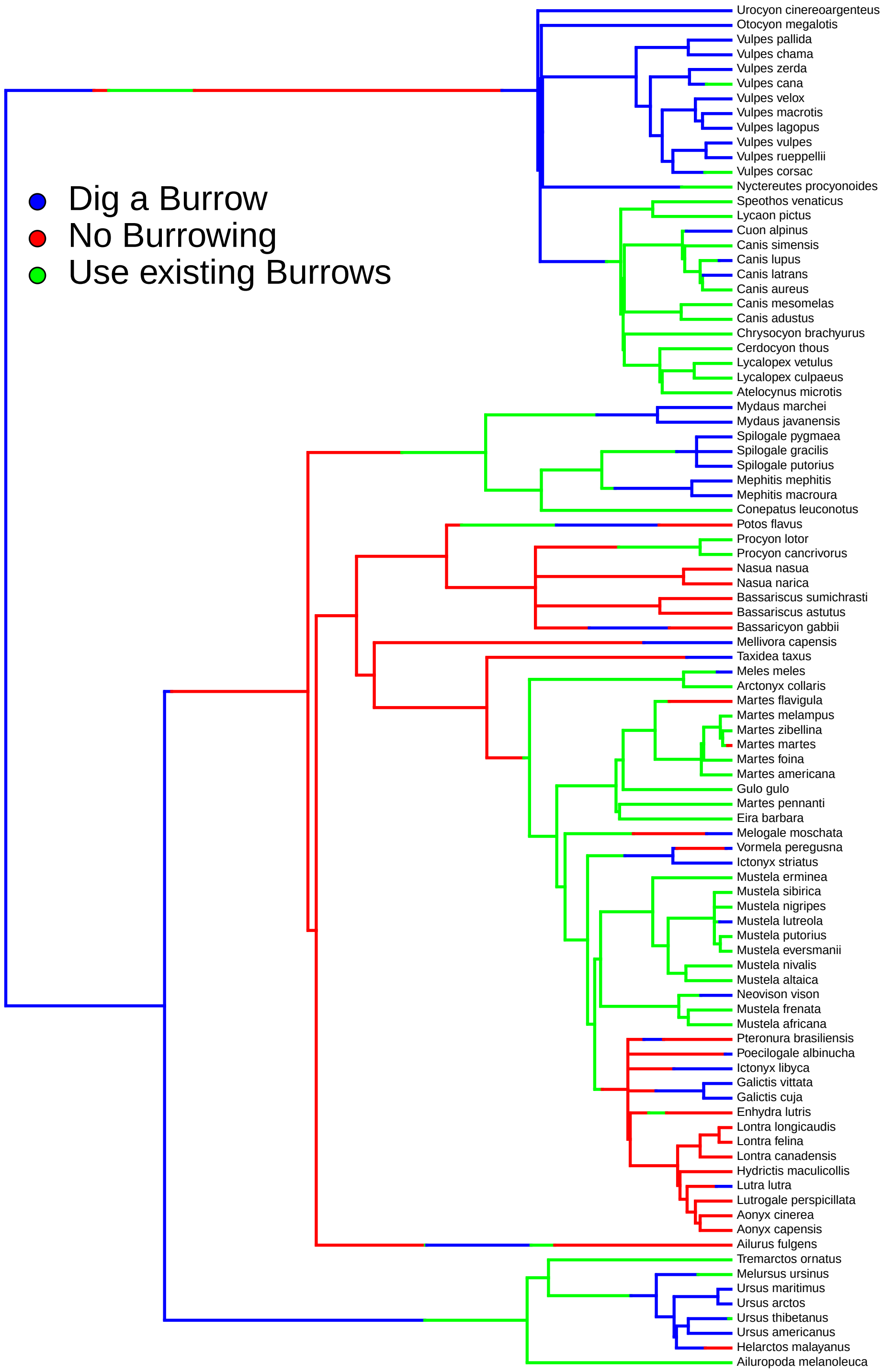
```
make.simmap is sampling character histories conditioned on the transition matrix

Q =
                     Dig a Burrow No Burrowing Use existing Burrows
Dig a Burrow          -0.05640412   0.02820206           0.02820206
```

```
No Burrowing           0.02820206  -0.05640412            0.02820206
Use existing Burrows   0.02820206   0.02820206           -0.05640412
(estimated using likelihood);
and (mean) root node prior probabilities
pi =
        Dig a Burrow         No Burrowing Use existing Burrows
           0.3333333            0.3333333            0.3333333
```

```
Done.
```

```r
cols <- setNames(c("blue", "red", "green"), sort(unique(burrow)))
plotSimmap(scm3, cols, pts = FALSE, lwd = 2, fsize = 0.5)
add.simmap.legend(colors = cols, vertical = TRUE, prompt = FALSE, x = 2, y = 80, fsize
```


Dig a Burrow
No Burrowing
Use existing Burrows
Urocyon cinereoargenteus
Otocyon megalotis
Vulpes pallida
Vulpes chama
Vulpes zerda
Vulpes cana
Vulpes velox
Vulpes macrotis
Vulpes lagopus
Vulpes vulpes
Vulpes rueppellii
Vulpes corsac
Nyctereutes procyonoides
Speothos venaticus
Lycaon pictus
Cuon alpinus
Canis simensis
Canis lupus
Canis latrans
Canis aureus
Canis mesomelas
Canis adustus
Chrysocyon brachyurus
Cerdocyon thous
Lycalopex vetulus
Lycalopex culpaeus
Atelocynus microtis
Mydaus marchei
Mydaus javanensis
Spilogale pygmaea
Spilogale gracilis
Spilogale putorius
Mephitis mephitis
Mephitis macroura
Conepatus leuconotus
Potos flavus
Procyon lotor
Procyon cancrivorus
Nasua nasua
Nasua narica
Bassariscus sumichrasti
Bassariscus astutus
Bassaricyon gabbii
Mellivora capensis
Taxidea taxus
Meles meles
Arctonyx collaris
Martes flavigula
Martes melampus
Martes zibellina
Martes martes
Martes foina
Martes americana
Gulo gulo
Martes pennanti
Eira barbara
Melogale moschata
Vormela peregusna
Ictonyx striatus
Mustela erminea
Mustela sibirica
Mustela nigripes
Mustela lutreola
Mustela putorius
Mustela eversmanii
Mustela nivalis
Mustela altaica
Neovison vison
Mustela frenata
Mustela africana
Pteronura brasiliensis
Poecilogale albinucha
Ictonyx libyca
Galictis vittata
Galictis cuja
Enhydra lutris
Lontra longicaudis
Lontra felina
Lontra canadensis
Hydrictis maculicollis
Lutra lutra
Lutrogale perspicillata
Aonyx cinerea
Aonyx capensis
Ailurus fulgens
Tremarctos ornatus
Melursus ursinus
Ursus maritimus
Ursus arctos
Ursus thibetanus
Ursus americanus
Helarctos malayanus
Ailuropoda melanoleuca

Let's sample 200 possible histories. This may take a few moments. For reports and publications, you should sample more than this. There's no hard rule but 1000 seems to be a good minimum for a proper analysis.

```
scm4 <- make.simmap(carn.tree, burrow, model = "ER", nsim = 200)
```

```
make.simmap is sampling character histories conditioned on the transition matrix

Q =
                     Dig a Burrow No Burrowing Use existing Burrows
Dig a Burrow          -0.05640412   0.02820206           0.02820206
No Burrowing           0.02820206  -0.05640412           0.02820206
Use existing Burrows   0.02820206   0.02820206          -0.05640412
(estimated using likelihood);
and (mean) root node prior probabilities
pi =
        Dig a Burrow         No Burrowing Use existing Burrows
           0.3333333            0.3333333            0.3333333

Done.
```

```
scm4.sum<-describe.simmap(scm4, plot = FALSE)
scm4.sum
```

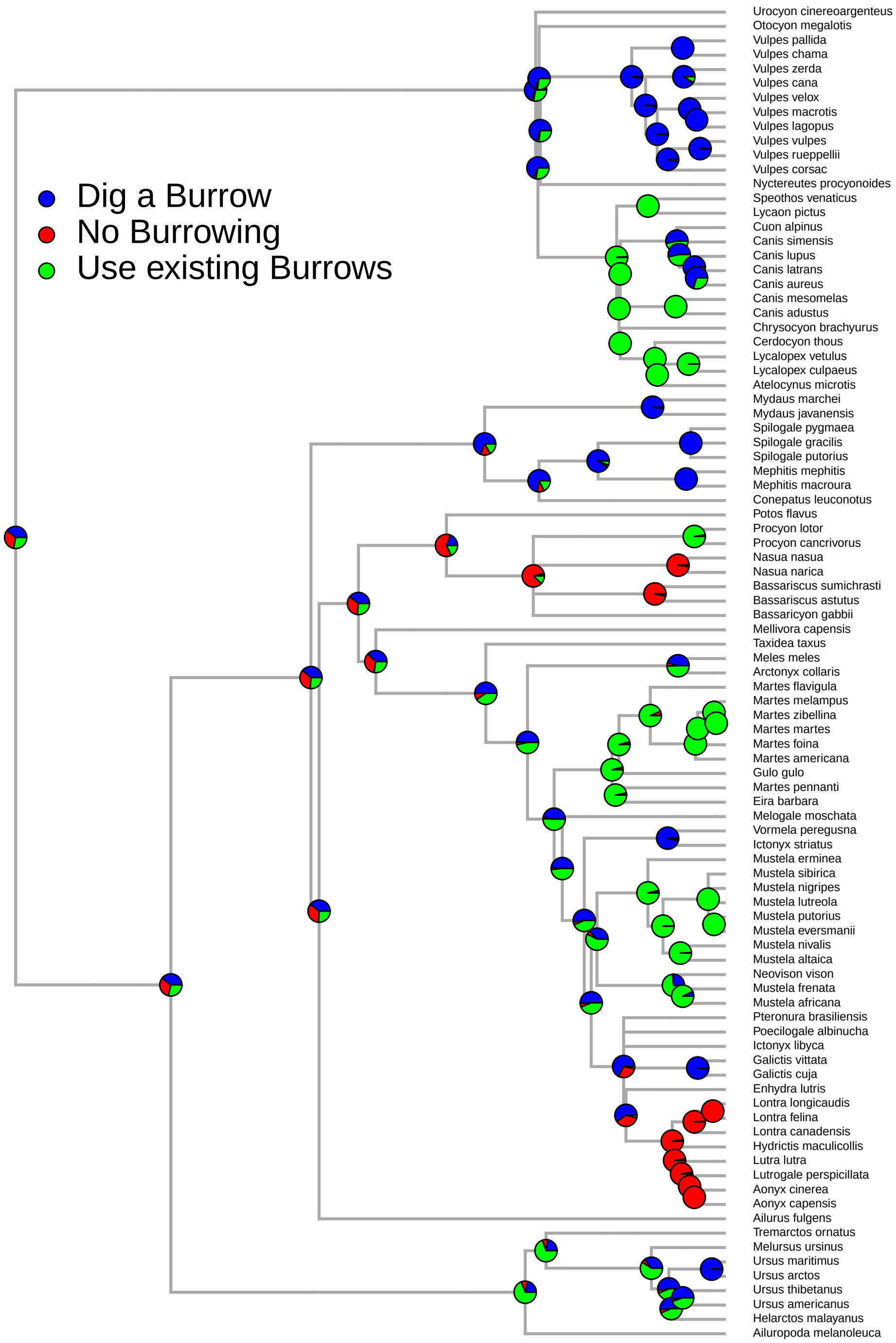
```
200 trees with a mapped discrete character with states:
 Dig a Burrow, No Burrowing, Use existing Burrows

trees have 52 changes between states on average

changes are of the following types:
     Dig a Burrow,No Burrowing Dig a Burrow,Use existing Burrows
x->y                      7.26                             12.24
     No Burrowing,Dig a Burrow No Burrowing,Use existing Burrows
x->y                     6.925                              5.99
     Use existing Burrows,Dig a Burrow Use existing Burrows,No Burrowing
x->y                            12.265                              7.32

mean total time spent in each state is:
     Dig a Burrow No Burrowing Use existing Burrows total
raw   366.1971509  214.5972284          350.2056207   931
prop    0.3933374    0.2305019            0.3761607     1
```

Finally we can plot the summary of the analysis as before.


Dig a Burrow
No Burrowing
Use existing Burrows
Urocyon cinereoargenteus
Otocyon megalotis
Vulpes pallida
Vulpes chama
Vulpes zerda
Vulpes cana
Vulpes velox
Vulpes macrotis
Vulpes lagopus
Vulpes vulpes
Vulpes rueppellii
Vulpes corsac
Nyctereutes procyonoides
Speothos venaticus
Lycaon pictus
Cuon alpinus
Canis simensis
Canis lupus
Canis latrans
Canis aureus
Canis mesomelas
Canis adustus
Chrysocyon brachyurus
Cerdocyon thous
Lycalopex vetulus
Lycalopex culpaeus
Atelocynus microtis
Mydaus marchei
Mydaus javanensis
Spilogale pygmaea
Spilogale gracilis
Spilogale putorius
Mephitis mephitis
Mephitis macroura
Conepatus leuconotus
Potos flavus
Procyon lotor
Procyon cancrivorus
Nasua nasua
Nasua narica
Bassariscus sumichrasti
Bassariscus astutus
Bassaricyon gabbii
Mellivora capensis
Taxidea taxus
Meles meles
Arctonyx collaris
Martes flavigula
Martes melampus
Martes zibellina
Martes martes
Martes foina
Martes americana
Gulo gulo
Martes pennanti
Eira barbara
Melogale moschata
Vormela peregusna
Ictonyx striatus
Mustela erminea
Mustela sibirica
Mustela nigripes
Mustela lutreola
Mustela putorius
Mustela eversmanii
Mustela nivalis
Mustela altaica
Neovison vison
Mustela frenata
Mustela africana
Pteronura brasiliensis
Poecilogale albinucha
Ictonyx libyca
Galictis vittata
Galictis cuja
Enhydra lutris
Lontra longicaudis
Lontra felina
Lontra canadensis
Hydrictis maculicollis
Lutra lutra
Lutrogale perspicillata
Aonyx cinerea
Aonyx capensis
Ailurus fulgens
Tremarctos ornatus
Melursus ursinus
Ursus maritimus
Ursus arctos
Ursus thibetanus
Ursus americanus
Helarctos malayanus
Ailuropoda melanoleuca

7.6 Further info

For more information about ancestral state reconstruction check out a review of the method by Joy *et al.* [Joy et al., 2016] and chapter 3 of *The comparative approach in evolutionary anthropology and biology* [Nunn, 2011].

For more information about the phytools package [Revell, 2012], the package author Liam Revell maintains an excellent blog here where you'll find lots of useful tips and demonstrations of the package's capabilities as well as some helpful troubleshooting.

Chapter 8

Ancestral State Reconstruction II

Previously, we looked at reconstructing the evolutionary history of binary traits, such as the presence or absence of sexual swellings in macaques, and categorical traits such as the modes of burrowing in carnivores. In this chapter, we'll be applying the same principles to continuous data.

The logic of ancestral state reconstruction applies equally to continuous traits like body size as it does to categorical traits. Here, we'll be looking at the evolutionary history of whales, dolphins and porpoises (Cetacea).

As always, check that you have set your working directory!

8.1 Data

The data we have here is taken from a study of the evolution of cetacean brain and body size [Montgomery et al., 2013]. The reduced version here contains only body mass and the log transformed body mass for 42 species.

```
whale.data <- read.table("whales_data.txt", header = T)
rownames(whale.data) <- whale.data$species
```

8.2 Tree

We also have a tree from the 10ktrees project [Arnold et al., 2010]. For more information about this website, see chapter 3.

```
whale.tree <- read.nexus("whales_tree.nex")
```

We need to check the data and tree match up. Get into this habit! It will save you a lot of time and patience.

```
rownames(whale.data) <- whale.data$species
geiger::name.check(whale.tree, whale.data)
```

```
$tree_not_data
 [1] "Balaenoptera_acutorostrata" "Balaenoptera_bonaerensis"
 [3] "Balaenoptera_edeni"         "Berardius_arnuxii"
 [5] "Berardius_bairdii"          "Caperea_marginata"
 [7] "Cephalorhynchus_eutropia"   "Cephalorhynchus_hectori"
 [9] "Delphinus_capensis"         "Delphinus_tropicalis"
[11] "Eubalaena_australis"        "Eubalaena_glacialis"
[13] "Eubalaena_japonica"         "Feresa_attenuata"
[15] "Hyperoodon_ampullatus"      "Hyperoodon_planifrons"
[17] "Indopacetus_pacificus"      "Lagenodelphis_hosei"
[19] "Lagenorhynchus_australis"   "Lagenorhynchus_cruciger"
[21] "Lissodelphis_peronii"       "Mesoplodon_bidens"
[23] "Mesoplodon_bowdoini"        "Mesoplodon_carlhubbsi"
[25] "Mesoplodon_ginkgodens"      "Mesoplodon_grayi"
[27] "Mesoplodon_hectori"         "Mesoplodon_layardii"
[29] "Mesoplodon_perrini"         "Mesoplodon_peruvianus"
[31] "Mesoplodon_stejnegeri"      "Orcaella_brevirostris"
[33] "Orcaella_heinsohni"         "Peponocephala_electra"
[35] "Phocoena_dioptrica"         "Phocoena_sinus"
[37] "Platanista_minor"           "Sousa_chinensis"
[39] "Stenella_attenuata"         "Stenella_frontalis"
[41] "Tasmacetus_shepherdi"       "Tursiops_aduncus"

$data_not_tree
character(0)
```

Clearly some species need to be dropped from the tree!

```
whale.tree <- drop.tip(whale.tree,
                       geiger::name.check(whale.tree, whale.data)$tree_not_data)
geiger::name.check(whale.tree, whale.data)
```

```
[1] "OK"
```

8.3 Ancestral State Reconstructions

Now we're going to dive in with a reconstruction. We are using **phytools** for this analysis so we should load the package and create a named data vector

[Revell, 2012].

```
require(phytools)
x <- whale.data$log.body.mass
names(x) <- whale.data$species
```

The function we need is called **fastAnc** and it returns the ancestral states in a simple list.

```
ancstates <- fastAnc(tree = whale.tree,    #Our phylogeny
                     x,                    #Our data vector
                     CI = TRUE)            #Estimate 95% confidence intervals
ancstates
```

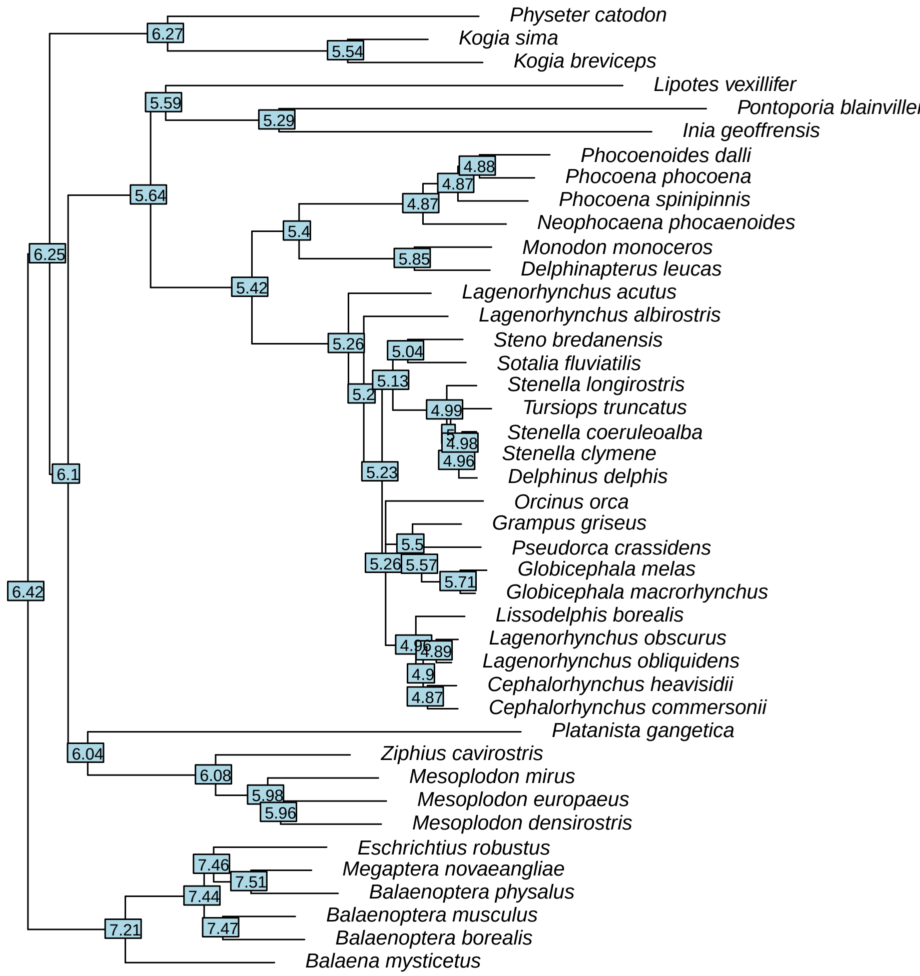
```
Ancestral character estimates using fastAnc:
      43       44       45       46       47       48       49       50
6.422936 7.205471 7.440591 7.465284 7.456463 7.511707 6.248774 6.097254
      51       52       53       54       55       56       57       58
6.044864 6.078069 5.983733 5.962172 5.641527 5.423179 5.255812 5.204571
      59       60       61       62       63       64       65       66
5.225028 5.260263 4.960018 4.901903 4.871755 4.888973 5.503471 5.567067
      67       68       69       70       71       72       73       74
5.710190 5.130854 4.989830 5.001555 4.960103 4.976035 5.039423 5.403331
      75       76       77       78       79       80       81       82
5.850745 4.870560 4.871380 4.883752 5.590679 5.292187 6.267476 5.540614

Lower & upper 95% CIs:
      lower    upper
43 5.745860 7.100012
44 6.599516 7.811426
45 7.009526 7.871657
46 7.037792 7.892775
47 7.023487 7.889440
48 7.078101 7.945312
49 5.617637 6.879912
50 5.480639 6.713870
51 5.396996 6.692732
52 5.495449 6.660689
53 5.489021 6.478445
54 5.474786 6.449558
55 4.977189 6.305864
56 4.818460 6.027899
57 4.833079 5.678546
58 4.834286 5.574857
59 4.930739 5.519317
60 4.970172 5.550353
61 4.679687 5.240349
```

```
62 4.656292 5.147513
63 4.621817 5.121692
64 4.661301 5.116645
65 5.175817 5.831126
66 5.231290 5.902844
67 5.455762 5.964619
68 4.810518 5.451190
69 4.731256 5.248405
70 4.760819 5.242292
71 4.767162 5.153044
72 4.800446 5.151624
73 4.692067 5.386779
74 4.792073 6.014589
75 5.352644 6.348847
76 4.380205 5.360915
77 4.449659 5.293101
78 4.482168 5.285335
79 4.892863 6.288496
80 4.417990 6.166384
81 5.492347 7.042604
82 4.959783 6.121445
```

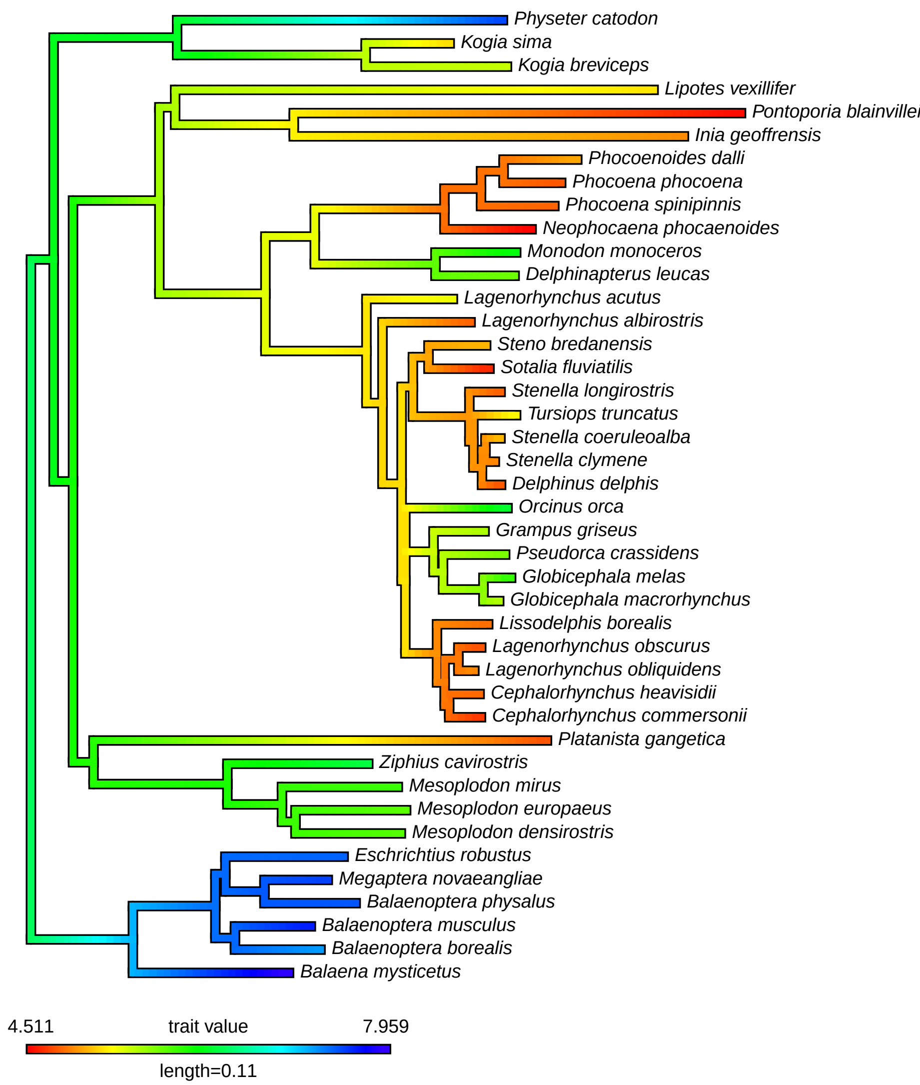
To get an idea of what these results show, we should probably plot it. The **nodelabels** function maps the ancestral states listed in our **ancstates** object onto the nodes of the tree which are listed in the same order.

```
plot(whale.tree, cex = .8, label.offset = .01, no.margin = TRUE)
nodelabels(round(ancstates$ace, digits = 2), cex = .67)
```



As is often the case, there are better ways to plot this information! The function **contMap** calls **fastAnc** and then maps the history of the trait onto the tree as a heatmap. This is a much clearer plot.

```
contMap(whale.tree, x, fsize = .7)
```



8.4 BayesTraits

Simply reconstructing the history of a trait can be very interesting. See some papers by Montgomery *et al.* [2010; 2013] for just a few great examples. However, this methodology is not limited to simply estimating the past.

Most of what we are going to do here could probably be acheived in R either with existing packages or some clever coding. However, the standard package for several analyses has been BayesTraits for some time.

BayesTraits is a command line program, which can make it kind of intimidating. Actually (like R), it's relatively easy to use but can take some getting used to. Fortunately, Randi Griffin has written an excellent R package **btw** that can operate the program from within R.

It's worth noting at this point that **btw** is not written to run BayesTraits for you so that you don't have to understand the program. Randi states very clearly that the package is purely for optimising workflow. In other words, this allows you to have all your data, results and code in one place. You still need to understand how to use the program. Fortunately the manual is very detailed.

First up, download BayesTraits version 3 for your operating system.

IMPORTANT! BayesTraits output files will be written into your working directory. They will overwrite any files with the same name so don't have any files called "data.txt", "tree.nex" or "inputfile.txt" in this directory unless you are ok with losing them.

Next, we need to install **btw**. This isn't a CRAN archived package so we'll be installing directly from Randi Griffin's GitHub. Once installed, we can use BayesTraits from within R!

```r
install.packages("devtools")
library(devtools)
install_github("rgriff23/btw")
library(btw)
```

There are some important differences in how R and BayesTraits read data that need to be summarised here.

- The first column of your data must contain species names.
- Species names must match exactly between tree and data (but don't worry about the order).
- No spaces in species names.
- Discrete characters have to be of class character or factor (between 0-9) and NOT integer.
- Ambiguous discrete characters can be represented as 01.
- Missing data must be represented as - rather than NA.

BayesTraits consists of modules (see manual for details) that are numbered and can be called up for different analyses.

If you can't get R and Bayestraits to play nicely together, you may want to consider using Bayestraits directly from the command prompt (Windows) or terminal (Mac). It's fairly straightforward once you've got the hang of it so be patient. Alternatively, all of this can be done with R packages like ape [Paradis and Schliep, 2018], geiger [Harmon et al., 2008] and phytools [Revell, 2012] amongst others.

8.5 Modelling Evolution

If we have some data about traits across a group of animals and an associated tree, we may want to ask about how that trait has evolved over time. For this we can compare the trait to models of evolutionary change.

8.5.1 Brownian Motion

Brownian motion (BM) is the most commonly used model of evolutionary change. In some ways, it can represent a kind of *null model* but do not confuse this! It doesn't mean nothing is changing or that evolution is not taking place.

Brownian motion assumes three things;

- Evolutionary changes in a trait are randomly distributed around a mean of 0.
- Evolutionary changes in a trait are independent of previous changes and changes on other branches.
- Larger changes are more likely to occur on longer branches.

All this means that BM is a *random walk* model in which the trait varies along the branches essentially at random.

We can use BayesTraits (via R) to model the evolution of body size in cetaceans with the assumption of Brownian motion. First we need to isolate our variables into a data table for **btw**. The way to do this is quite simple. We can simply extract the two columns we need (1 and 2) into a new object.

```
BT.data <- whale.data[,c(1,2)]
rownames(BT.data) <- NULL
head(BT.data)
```

```
                species log.body.mass
1       Kogia_breviceps      5.523746
2            Kogia_sima      5.226600
3      Physeter_catodon      7.573065
4  Platanista_gangetica      4.775465
5 Delphinapterus_leucas      5.803457
6     Monodon_monoceros      6.198198
```

This first analysis corresponds to **Continuous: Random Walk Model A ML** in the BayesTraits manual. We can see from the manual that the commands to run this are “4 1 Run”. You need to be familiar with BayesTraits to interpret this so the first time you do it, you may want to do it in BayesTraits directly (via the command prompt or terminal). In essence, BayesTraits asks us questions and provides us with options for what we want it to do and **4, 1, Run** are the options to run this analysis.

Given that we know what we want to do ahead of time, we can enter the commands into a command vector in R. To run these commands through BayesTraits, R will write them into a text file so BayesTraits can interpret them when needed. Note that you don't need to enter **Run** into this vector as **btw** will take care of that for us.

```
command_vec1 <- c("4", "1")
```

Note that if you have nodelabels in your tree, there will be an error when running BayesTraits. You can remove nodelabels without effecting the structure of your tree like this.

```
whale.tree$node.label <- NULL
```

I have a path on my desktop just for BayesTraits analyses. Remember that there must be a copy of BayesTraitsV3 stored here. That's all you need as the output will be read back into R by **btw**. You also should remember to change your working directory back if you are finished with BayesTraits. In this chunk, I've saved the existing directory at the start and reset it immediately after the analysis is completed.

```
wd.reset <- getwd()
setwd("~/Desktop/BayesTraits")
m1 <- bayestraits(data = BT.data, tree = whale.tree, commands = command_vec1)
setwd(wd.reset)
```

On we go! The object that should have appeared in your R environment contains all the outputs you need from BayesTraits. Let's have a look at the **results** component of the **Log**.

```
m1$Log$results
```

```
  Tree.No       Lh  Alpha.1 Sigma.2.1
1       1 -31.9823 6.422936  6.315406
```

These results give us the Log likelihood (**Lh**), the reconstructed ancestral node (**Alpha.1**) and the phylogenetically corrected variance of the data (**Sigma.2.1**). The important thing to look at here is the log likelihood. We will use that to compare the BM model to other models.

8.5.2 Directional Evolution

So far we've looked at the random walk model of evolution. In reality, what we are usually interested in is deviations from the random walk model. We can investigate this using similar methods, but with a **directional** model.

An example of a case when we might be interested in a directional model is Cope's rule [Kingsolver and Pfennig, 2004, Hone and Benton, 2005]. Cope's rule states that over time, lineages tend to have larger body sizes. So basically, on average animals tend to get bigger over evolutionary time.

Let's see if we can detect a trend in cetacean body mass. For this analysis, we need a non-ultrametric tree (a phylogram rather than a chronogram). Luckily that's what we already have. The branch lengths here describe evolutionary distance in terms of genetic change and so shorter branches indicate fewer genetic changes.

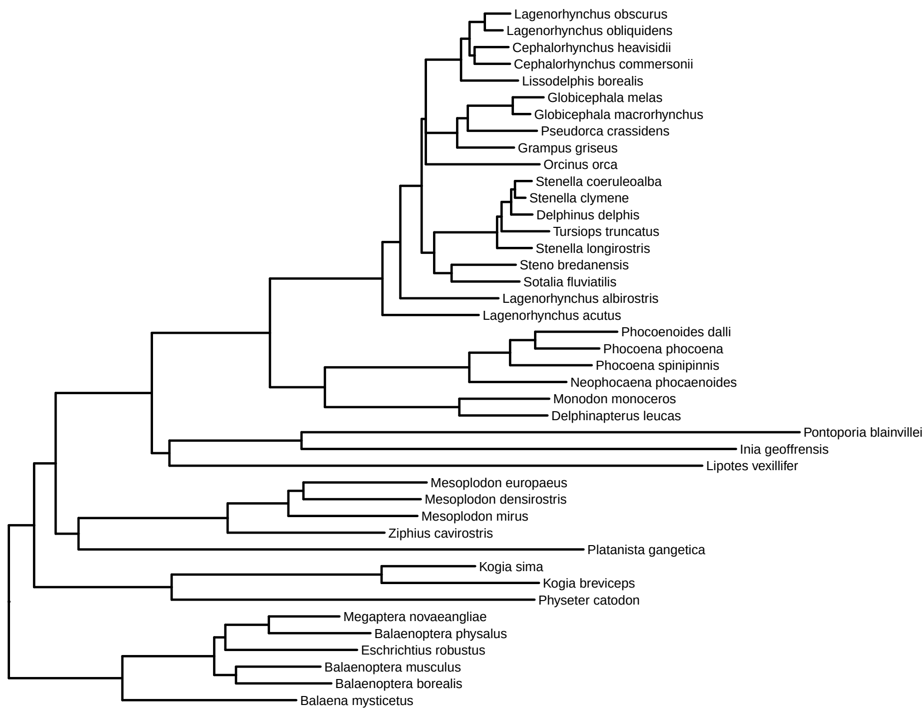


Figure 8.1: Phylogenetic tree of 42 species of cetcaeans with branch lengths proportional to molecular change.

We're using BayesTraits again so the first step is to get our data into the right format.

```
BT.data <- whale.data[,c(1,2)]
rownames(BT.data) <- NULL
```

As before, we run the random walk (BM) model first. Remember we need to set the working directory to a path where BayesTraits is stored.

```
setwd("~/Desktop/BayesTraits")
RW.commands <- c("4", "1")
RWmod <- bayestraits(BT.data, whale.tree, RW.commands)
```

```r
RWmod$Log$results
```

```
  Tree.No       Lh  Alpha.1 Sigma.2.1
1       1 -31.9823 6.422936  6.315406
```

```r
setwd(wd.reset)
```

The directional model takes a different set of commands.

```r
setwd("~/Desktop/BayesTraits")
D.commands <- c("5", "1")
Dmod <- bayestraits(BT.data, whale.tree, D.commands)
Dmod$Log$results
```

```
  Tree.No        Lh  Alpha.1    Beta.1 Sigma.2.1
1       1 -30.22462 7.951217 -12.37922  5.808331
```

```r
setwd(wd.reset)
```

Now we need to compare these models! What he have so far is two models and a log likelihood assigned to each. This means we can compare them using a **likelihood ratio test**. The general formula for an LR test is;

$$LR = 2 * (Lh_{ModelB} - Lh_{ModelA})$$

The result is the **likelihood ratio statistic** (LR) which is asymptotically χ^2 distributed with degrees of freedom equal to the difference in the number of parameters between the models. Model A has 1 parameter (the root value) and model B has 2 (the root and the direction of change) so the degrees of freedom are 1.

```r
2*(Dmod$Log$results$Lh[1] - RWmod$Log$results$Lh[1])
```

```
[1] 3.515352
```

```r
1-pchisq(3.515352, df = 1)
```

```
[1] 0.06080274
```

Note: pchisq gives the proportion of the distribution to the left of the value. To test if the model is better than the null model, we use 1 - pchisq.

The **btw** package has a function that will do all this for us. Be careful with interpretation though. Note that the p-value is different. Take this away from 1 and you have your p-value as above.

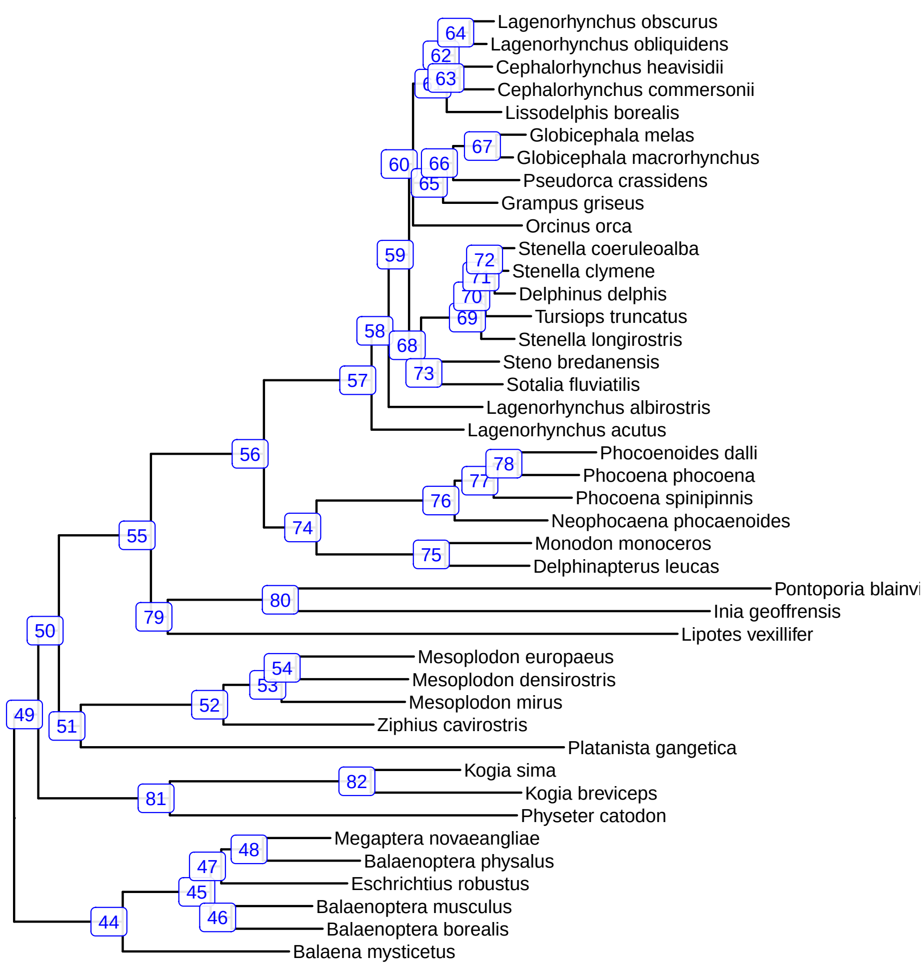
```r
lrtest(RWmod, Dmod)
```

```
  model1.Lh model2.Lh   LRstat      pval
1  -31.9823 -30.22462 3.515352 0.9391973
```

So what have we got here? Well we have tested two models of the evolution of body size in cetacea. The first is a random walk (Brownian motion) model of evolution in which we have estimated two parameters. The second is a directional model in which we have estimated 3 parameters. Model comparison showed no significant difference between them (LR = 3.52, p = 0.06) and so we should favour the simpler, 2 parameter model. Thus we have no evidence for a directional trend in cetacean body mass evolution.

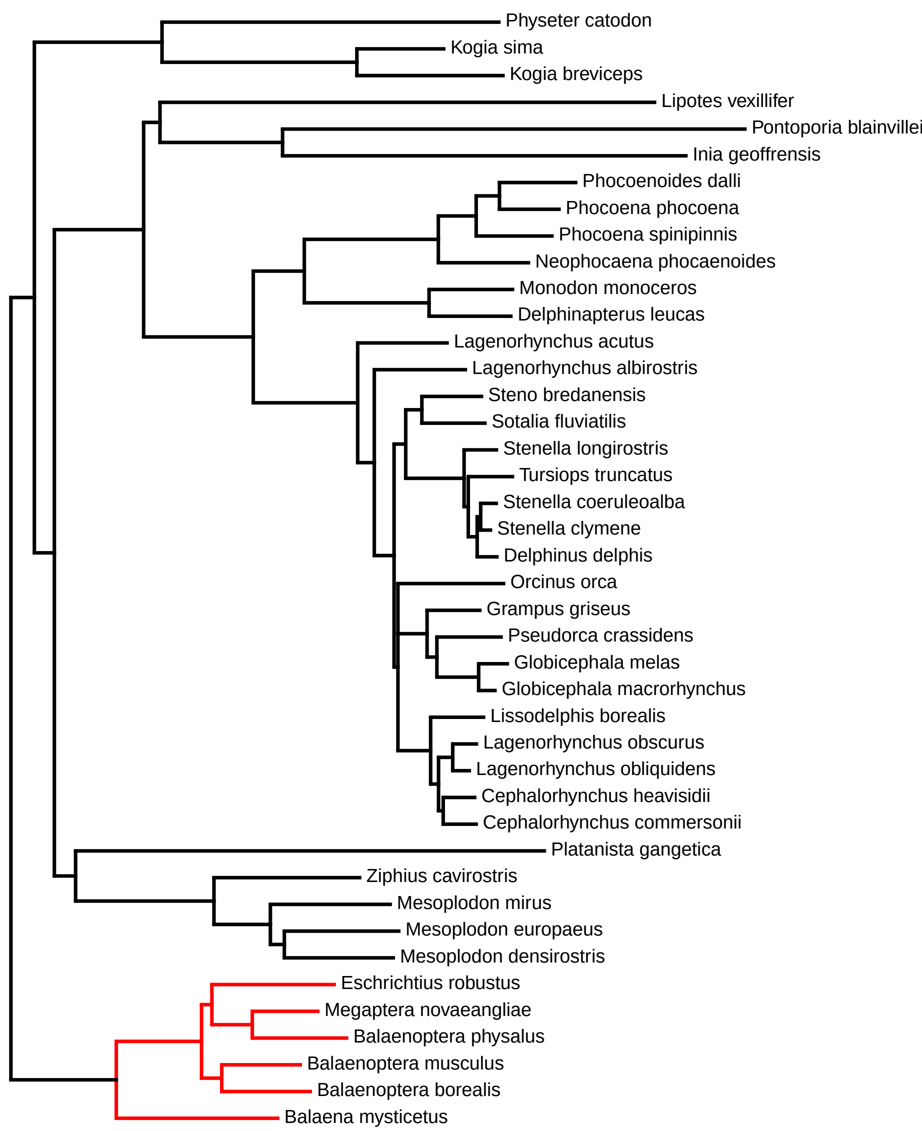
8.6 Changes in the rate of evolution of a trait

Often when investigating the evolution of a continuous trait, we might have reason to suggest that in some lineages, the rate of evolution of that trait changed.



Let's say we have a hypothesis that says the rate of change of body mass changed at the root of mysticetes (node 44). We can paint that onto the tree for demonstration using **paintSubTree** and **plotSimmap** in **phytools** [Revell, 2012].

```
require(phytools)
tree1 <- paintSubTree(whale.tree, 44, "2")
plotSimmap(tree1, lwd = 2, fsize = 0.7)
```



Now we can run the test. Here the function **brownie.lite** in **phytools** compares the single rate model to the multi-rate model we have specified!

```
x <- whale.data$log.body.mass
names(x) <- whale.data$species
fit <- brownie.lite(tree1, x)
fit
```

```
ML single-rate model:
    s^2 se  a   k   logL
value   6.165   1.3453  6.4229  2   -31.9823

ML multi-rate model:
    s^2(1)  se(1)   s^2(2)  se(2)   a   k   logL
value   7.1119  1.6786  1.2663  0.8687  6.6116  3   -30.475

P-value (based on X^2): 0.0825

R thinks it has found the ML solution.
```

Here we've found no evidence of a regime shift in mysticete cetaceans ($p = 0.083$).

8.7 Uncertainty

If you are familiar with cetaceans and their evolutionary history, you might be surprised by our findings so far in this chapter. The prevailing state of knowledge suggests that cetaceans have evolved large body sizes since the transition to the water of an approximately dog-sized ancestor at the root of our tree. Given what we know about the fossil record of cetacea, we would expect to detect an increase in body size over the tree. To solve this puzzle, we need to look at what information we provided our analysis with.

As the old saying goes, *if you put garbage in, you'll get garbage out* and this seems to apply here. For example, let's look closely at our reconstructions. You can see here that both reconstructions have estimated the mass of the ancestor of cetaceans. Remember that these are log transformed data so we have to transform them back if we want to get a straightforward measurement of mass.

```
10^(RWmod$Log$results$Alpha.1)
```

```
[1] 2648111
```

```
10^(Dmod$Log$results$Alpha.1)
```

```
[1] 89375216
```

So depending on our model of evolution the ancestor was either 2,648.1 kg or 89,375.2 kg. A big difference between models so which one we choose really matters.

This is even more of a problem when we look at the fossil record of cetaceans. *Indohyus* (Raoellidae) is thought to be the species that most closely represents the transition to the water by cetacean ancestors [Thewissen et al., 2009] and its mass is estimated at around 10kg. An early species of cetacean called *Pakicetus* was estimated at around 45kg. So we are orders of magnitude away from what the fossil record shows us!

This problem is well understood in phylogenetic comparative methods. In fact, all methods of ancestral state reconstruction perform very poorly when compared to what we know from the fossil record [Webster and Purvis, 2002]. As you might expect, the deeper into your tree you try to estimate an ancestral state, the greater the uncertainty. This is especially clear when you look at estimating the root [Gascuel and Steel, 2014]. The solution is to incorporate fossil data in the analysis [Slater et al., 2012].

8.7.1 Fossils

To demonstrate the importance of fossil data, let's take a closer look at the evolution of body size in cetaceans. With **fastAnc**, we found a mass of around 2,650kg for the root of the cetaceans.

The package **RRphylo** [Castiglione et al., 2018] contains data on fossil and living cetaceans [Serio et al., 2019]. Using these data, we can hopefully perform a more rigorous ancestral state reconstruction [Castiglione et al., 2020]. Note that the values here differ betwwen datasets because the previous dataset used a log10 transformation whereas this one uses a natural log transformation!

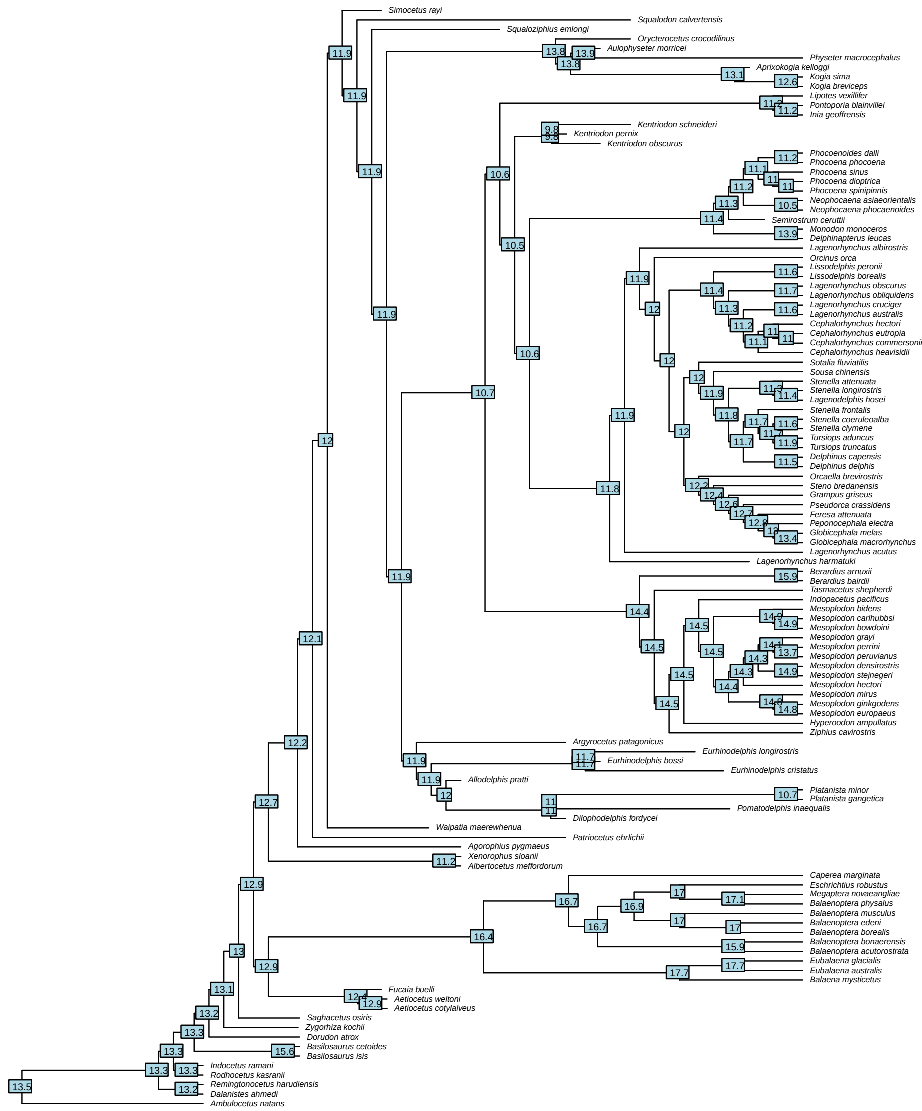
```
library(RRphylo)
data("DataCetaceans")
DataCetaceans$treecet -> treecet
DataCetaceans$masscet -> masscet
```

The **RRphylo** function performs a variant of ancestral state reconstruction called **phylogenetic ridge regression** [Castiglione et al., 2020].

```
RR <- RRphylo(treecet, masscet)
```

RRphylo returns a lot of information as a list. Included in this list is the **tree** used (useful for plotting) and **aces** which contains the estimates for the traits at the nodes.

```
plot(RR$tree, cex = .4, label.offset = .5, no.margin = TRUE)
nodelabels(round(RR$aces, digits = 1), cex = .5)
```



Using this reconstruction, we can extract the mass of the root. Remember that we need **exp** to calculate the untransformed value rather than raising to the power of 10 because of the natural log transformation.

```
exp(RR$aces[[1]])
```

```
[1] 727063.9
```

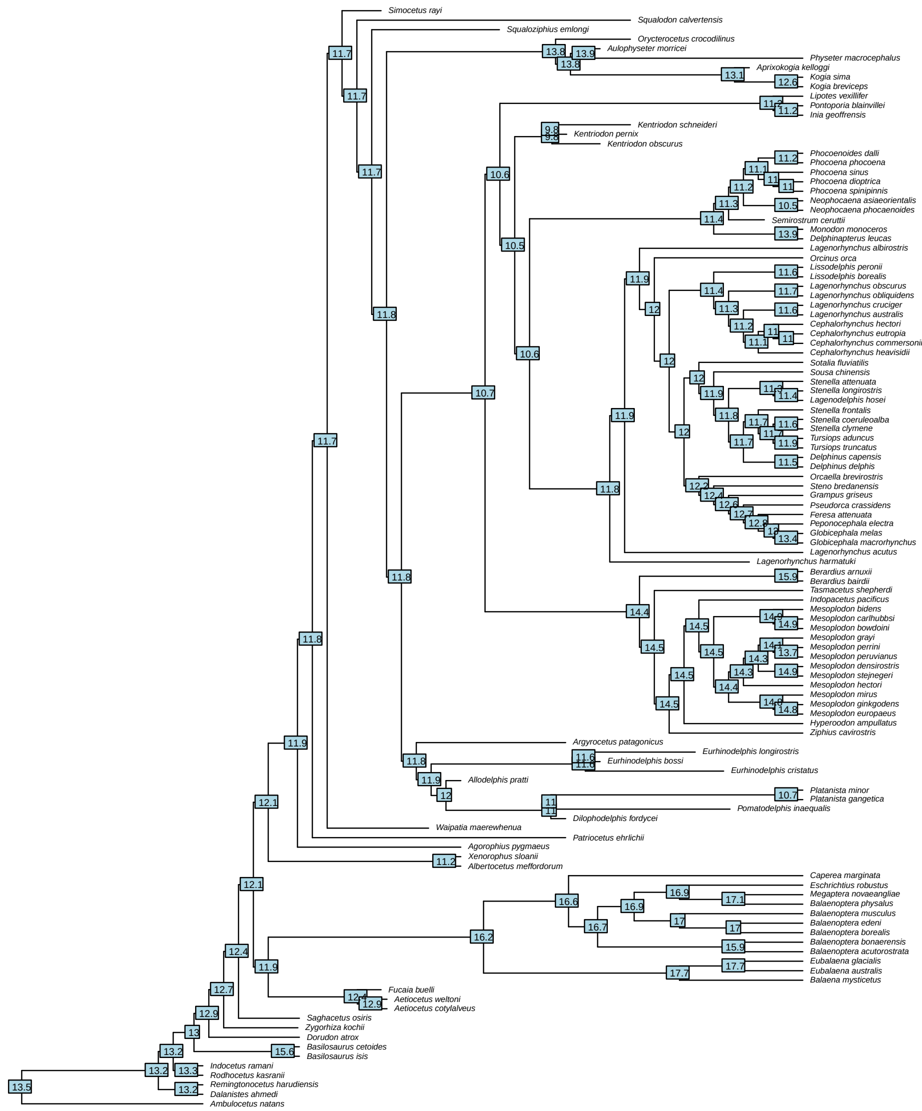
So our new estimate of the mass of the ancestor of cetaceans is 727.1kg. This is much closer to the estimated mass of early archeocete cetaceans like *Ambulocetus natans* at about 430kg and *Indocetus ramani* at around 630kg.

If we still aren't satisfied that we have included the best information we have available, we can actually *fossilise* a node by passing a named list of ancestral states to the **RRphylo** function. Following the example of Castiglione *et al.* [2020], we can set the node of the ancestor of mysticetes to a known mass. Here we are assuming that the most recent common ancestor of all mysticetes can be represented by the species *Mystacodon selenensis* which weighed arond 150kg. Also we need to know that this ancestor is represented by the node labelled 128 in our tree object.

```
x <- log(150000)
names(x) <- "128"
```

Now we can pass this state to the argument **aces** in **RRphylo** and the analysis will hold node 128 at the value we have set. You should be able to see that in the following plot, the ancestor of mysticetes is reconstructed as 11.9 rather than 12.9 in the previous reconstruction.

```
RR2 <- RRphylo(treecet, masscet, aces = x)
plot(RR2$tree, cex = .4, label.offset = .5, no.margin = TRUE)
nodelabels(round(RR2$aces, digits = 1), cex = .5)
```



Hopefully you can see that the more fossil information you include in your reconstructions, the more reliable they are.

8.7.2 Revisiting Mysticete Body Mass

In using the fossil data we have added in here, Castiglione *et al.* [2020] demonstrated that mysticetes actually do conform to Cope's rule because they have an increasing trend in body size over time. This shows just how important adding in fossil data can be if you want the full picture.

This seems to suggest that we should also find a regime shift in mysticetes. Let's have a closer look. We begin again by painting the tree at the specific node leading to mysticetes.

```
require(phytools)
tree2 <- paintSubTree(treecet, 128, "2")
plotSimmap(tree2, lwd = 2, fsize = 0.5)
```

Next we run **brownie.lite** on our expanded dataset.

```
fit <- brownie.lite(tree2, masscet)
fit
```

```
ML single-rate model:
    s^2 se  a   k   logL
value   0.2047  0.0265  13.2057 2   -178.7922

ML multi-rate model:
    s^2(1)  se(1)   s^2(2)  se(2)   a   k   logL
value   0.176   0.0245  0.4012  0.147   13.2056 3   -176.0958

P-value (based on X^2): 0.0202

R thinks it has found the ML solution.
```

There you have it! We can now say that we have evidence in favour of a regime shift in mysticete body size ($p = 0.02$).

8.8 Further info

We've only just scratched the surface of what is possible with ancestral state reconstruction. For some background reading, have a look at chapter 4 of *The comparative approach in evolutionary anthropology and biology* [Nunn, 2011].

Chapter 9

Bibliography

Bibliography

Christian Arnold, Luke J. Matthews, and Charles L. Nunn. The 10ktrees website: A new online resource for primate phylogeny. *Evolutionary Anthropology: Issues, News, and Reviews*, 19(3):114–118, 2010. doi: 10.1002/evan.20251. URL https://doi.org/10.1002/evan.20251.

D.A. Baum and S.D. Smith. *Tree Thinking: An Introduction to Phylogenetic Biology*. Macmillan Learning, 2012. ISBN 9781936221165. URL https://books.google.co.uk/books?id=zW_ApwAACAAJ.

Silvia Castiglione, Gianmarco Tesone, Martina Piccolo, Marina Melchionna, Alessandro Mondanaro, Carmela Serio, Mirko Di Febbraro, and Pasquale Raia. A new method for testing evolutionary rate variation and shifts in phenotypic evolution. *Methods in Ecology and Evolution*, 9:974–983, 2018. URL https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/2041-210X.12954.

Silvia Castiglione, Carmela Serio, Alessandro Mondanaro, Marina Melchionna, Francesco Carotenuto, Mirko Di Febbraro, Antonio Profico, Davide Tamagnini, and Pasquale Raia. Ancestral state estimation with phylogenetic ridge regression. *Evolutionary Biology*, 2020. doi: 10.1007/s11692-020-09505-x. URL https://doi.org/10.1007/s11692-020-09505-x.

Jr. Garland, Theodore, Allan W. Dickerman, Christine M. Janis, and Jason A. Jones. Phylogenetic Analysis of Covariance by Computer Simulation. *Systematic Biology*, 42(3):265–292, 1993. ISSN 1063-5157. doi: 10.1093/sysbio/42.3.265. URL https://doi.org/10.1093/sysbio/42.3.265.

Olivier Gascuel and Mike Steel. Predicting the ancestral character changes in a tree is typically easier than predicting the root state. *Systematic Biology*, 63 (3):421–435, 2014. doi: 10.1093/sysbio/syu010. URL https://doi.org/10.1093/sysbio/syu010.

LJ Harmon, JT Weir, CD Brock, RE Glor, and W Challenger. Geiger: investigating evolutionary radiations. *Bioinformatics*, 24:129–131, 2008.

David W. E. Hone and Michael J. Benton. The evolution of large size: how does cope's rule work? *Trends in Ecology & Evolution*, 20(1):4–6, 2005. URL https://liverpool.idm.oclc.org/login?url=https://search.ebscohost.c om/login.aspx?direct=true&db=edselp&AN=S0169534704003143&site=eds-live&scope=site.

Nizar Ibrahim, Simone Maganuco, Cristiano Dal Sasso, Matteo Fabbri, Marco Auditore, Gabriele Bindellini, David M. Martill, Samir Zouhri, Diego A. Mattarelli, David M. Unwin, Jasmina Wiemann, Davide Bonadonna, Ayoub Amane, Juliana Jakubczak, Ulrich Joger, George V. Lauder, and Stephanie E. Pierce. Tail-propelled aquatic locomotion in a theropod dinosaur. *Nature*, 581(7806):67–70, 2020. doi: 10.1038/s41586-020-2190-3. URL https://doi.org/10.1038/s41586-020-2190-3.

Jeffrey B. Joy, Richard H. Liang, Rosemary M. McCloskey, T. Nguyen, and Art F. Y. Poon. Ancestral reconstruction. *PLOS Computational Biology*, 12(7): e1004763–, 2016. URL https://doi.org/10.1371/journal.pcbi.1004763.

Joel G. Kingsolver and David W. Pfennig. Individual-level selection as a cause of cope's rule of phyletic size increase. *Evolution*, 58(7):1608, 2004. URL https://liverpool.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edsjsr&AN=edsjsr.3449385&site=eds-live&scope=site.

Michael R. McGowen, Stephen H. Montgomery, Clay Clark, and John Gatesy. Phylogeny and adaptive evolution of the brain-development gene microcephalin (mcph1) in cetaceans. *BMC Evolutionary Biology*, 11(1):98, 2011. doi: 10.1186/1471-2148-11-98. URL https://doi.org/10.1186/1471-2148-11-98.

Stephen H. Montgomery, Isabella Capellini, Robert A. Barton, and Nicholas I. Mundy. Reconstructing the ups and downs of primate brain evolution: implications for adaptive hypotheses and homo floresiensis. *BMC Biology*, 8(1):9, 2010. doi: 10.1186/1741-7007-8-9. URL https://doi.org/10.1186/1741-7007-8-9.

Stephen H. Montgomery, Jonathan H. Geisler, Michael R. McGowen, Charlotte Fox, Lori Marino, and John Gatesy. The evolutionary history of cetacean brain and body size. 67(11):3339–3353, 2013. URL www.jstor.org/stable/24032748.

Martin Morgan. *BiocManager: Access the Bioconductor Project Package Repository*, 2019. URL https://CRAN.R-project.org/package=BiocManager. R package version 1.30.9.

C.L. Nunn. *The Comparative Approach in Evolutionary Anthropology and Biology*. University of Chicago Press, 2011. ISBN 9780226608983. URL https://books.google.co.uk/books?id=qj4cSzJGQJAC.

Katrin Nyakatura and Olaf RP Bininda-Emonds. Updating the evolutionary history of carnivora (mammalia): a new species-level supertree complete with divergence time estimates. *BMC Biology*, 10(1):12, 2012. doi: 10.1186/1741-7007-10-12. URL https://doi.org/10.1186/1741-7007-10-12.

E. Paradis and K. Schliep. ape 5.0: an environment for modern phylogenetics and evolutionary analyses in R. *Bioinformatics*, 35:526–528, 2018.

Heidi G. Parker, Dayna L. Dreger, Maud Rimbault, Brian W. Davis, Alexandra B. Mullen, Gretchen Carpintero-Ramirez, and Elaine A. Ostrander. Genomic analyses reveal the influence of geographic origin, migration, and hybridization on modern dog breed development. *Cell Reports*, 19(4):697 – 708, 2017. ISSN 2211-1247. doi: https://doi.org/10.1016/j.celrep.2017.03.079. URL http://www.sciencedirect.com/science/article/pii/S2211124717304564.

Ralph S. Peters, Lars Krogmann, Christoph Mayer, Alexander Donath, Simon Gunkel, Karen Meusemann, Alexey Kozlov, Lars Podsiadlowski, Malte Petersen, Robert Lanfear, Patricia A. Diez, John Heraty, Karl M. Kjer, Seraina Klopfstein, Rudolf Meier, Carlo Polidori, Thomas Schmitt, Shanlin Liu, Xin Zhou, Torsten Wappler, Jes Rust, Bernhard Misof, and Oliver Niehuis. Evolutionary history of the hymenoptera. *Current Biology*, 27(7):1013–1018, 2017. doi: 10.1016/j.cub.2017.01.027. URL https://doi.org/10.1016/j.cub.2017.01.027.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2019.

R Hackathon et al. *phylobase: Base Package for Phylogenetic Structures and Comparative Data*, 2019. URL https://CRAN.R-project.org/package=phylobase. R package version 0.8.6.

Simon M. Reader, Yfke Hager, and Kevin N. Laland. The evolution of primate general and cultural intelligence. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 366(1567):1017–1027, 2011. doi: 10.1098/rstb.2010.0342. URL https://royalsocietypublishing.org/doi/abs/10.1098/rstb.2010.0342.

Liam J. Revell. phytools: An r package for phylogenetic comparative biology (and other things). *Methods in Ecology and Evolution*, 3:217–223, 2012.

Carmela Serio, Silvia Castiglione, Gianmarco Tesone, Martina Piccolo, Marina Melchionna, Alessandro Mondanaro, Mirko Di Febbraro, and Pasquale Raia. Macroevolution of toothed whales exceptional relative brain size. *Evolutionary Biology*, 46:332–342, 2019. URL https://link.springer.com/article/10.1007/s11692-019-09485-7.

Graham J. Slater, Luke J. Harmon, and Michael E. Alfaro. Integrating fossils with molecular phylogenies improves inference of trait evolution. *Evolution*, 66(12):3931–3944, 2012. doi: 10.1111/j.1558-5646.2012.01723.x. URL https://doi.org/10.1111/j.1558-5646.2012.01723.x.

Theodore Stankowich, Tim Caro, and Matthew Cox. Bold coloration and the evolution of aposematism in terrestrial carnivores. *Evolution*, 65(11):3090, 2011. URL https://liverpool.idm.oclc.org/login?url=https://search.ebscoho st.com/login.aspx?direct=true&db=edsjsr&AN=edsjsr.41317030&site=eds-live&scope=site.

J. G. M. Thewissen, Lisa Noelle Cooper, John C. George, and Sunil Bajpai. From land to water: the origin of whales, dolphins, and porpoises. *Evolution: Education and Outreach*, 2(2):272–288, 2009. doi: 10.1007/s12052-009-0135-2. URL https://doi.org/10.1007/s12052-009-0135-2.

Andrea J. Webster and Andy Purvis. Testing the accuracy of methods for reconstructing ancestral states of continuous characters. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 269(1487):143–149, 2002. doi: 10.1098/rspb.2001.1873. URL https://doi.org/10.1098/rspb.2001.1873.

Hadley Wickham. *tidyverse: Easily Install and Load the 'Tidyverse'*, 2017. URL https://CRAN.R-project.org/package=tidyverse. R package version 1.2.1.

Guangchuang Yu. *ggimage: Use Image in 'ggplot2'*, 2019. URL https://CRAN.R-project.org/package=ggimage. R package version 0.2.4.

Guangchuang Yu, David Smith, Huachen Zhu, Yi Guan, and Tommy Tsan-Yuk Lam. ggtree: an r package for visualization and annotation of phylogenetic trees with their covariates and other associated data. *Methods in Ecology and Evolution*, 8:28–36, 2017. doi: 10.1111/2041-210X.12628. URL http://onlinelibrary.wiley.com/doi/10.1111/2041-210X.12628/abstract.

Guangchuang Yu, Tommy Tsan-Yuk Lam, Huachen Zhu, and Yi Guan. Two methods for mapping and visualizing associated data on phylogeny using ggtree. *Molecular Biology and Evolution*, 35:3041–3043, 2018. doi: 10.1093/molbev/msy194. URL https://doi.org/10.1093/molbev/msy194.