Phylogenetic Independent Contrasts in R

Casey Dunn

April 4, 2013

Contents

1 Introduction

This example analysis presents the basics phylogenetic independent contrasts (Felsenstein, 1985). It expands on the example presented at the R phylogenetics wiki (`http://www.r-phylo.org/wiki/HowTo/Phylogenetic_Independent_Contrasts`).

This analysis uses the programming language R. In the past decade, a large community of phylogenetic biologists interested in character evolution have written many analysis tools in R. This set of tools provides a rich environment for the study of character evolution.

2 Getting oriented with R

R, like python and many other languages, can be used interactively, where you enter a few commands at a time, or in batch mode, where a series of commands are placed in a file and executed all at once. We will use R interactively, and load R code that others have already written for phylogenetic analyses as we go.

There are already *many* general introductions, tutorials, and quick-references for R. I therefore won’t provide a background on R itself, we will dive right into some analyses.

There is detailed information on phylogenetic analysis with R at both the R phylo wiki and the CRAN task page for phylogenetics.

For the analyses below, you will need to install the ‘ape‘ library in R.

3 Getting set up and loading the data

Copy the data files (‘Geospiza.txt‘ and ‘Geospiza.nex‘) to a directory on your computer. Open the R interface (which was installed alongside the rest of the R components) or open a terminal window and launch an interactive R session with the command ‘R‘.

Now, change to the directory where your data files are:

```r
setwd("DATADIR")
```

Where ‘DATADIR‘ is the directory where you put the data files. The R command ‘setwd()‘ is much like the shell command ‘cd‘ - it controls your working directory.

Load the ape library, which has the functions we will use for phylogenetic independent contrasts:

```r
library(ape)
```

Load and prepare the data:

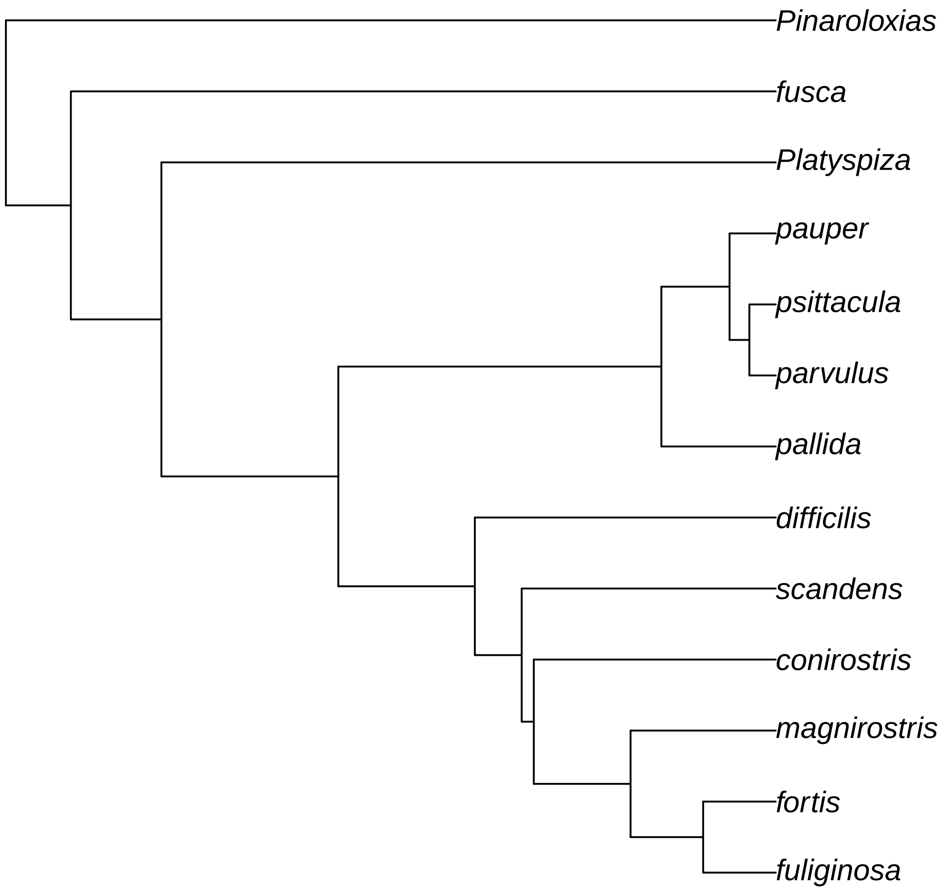
```r
geodata <- read.table("Geospiza.txt")
geotree <- read.nexus("Geospiza.nex")
```

We now need to remove a taxon in the tree that isn’t in the data file.

```r
geotree <- drop.tip(geotree, "olivacea")
```

Plot the tree:

```r
plot(geotree)
```



And take a look at the character data:

```
geodata

#               wingL tarsusL culmenL beakD gonysW
# magnirostris 4.404   3.039   2.725 2.824  2.676
# conirostris  4.350   2.984   2.654 2.514  2.360
# difficilis   4.224   2.899   2.277 2.011  1.930
# scandens     4.261   2.929   2.622 2.145  2.037
# fortis       4.244   2.895   2.407 2.363  2.222
# fuliginosa   4.133   2.807   2.095 1.941  1.845
```

Independent contrasts

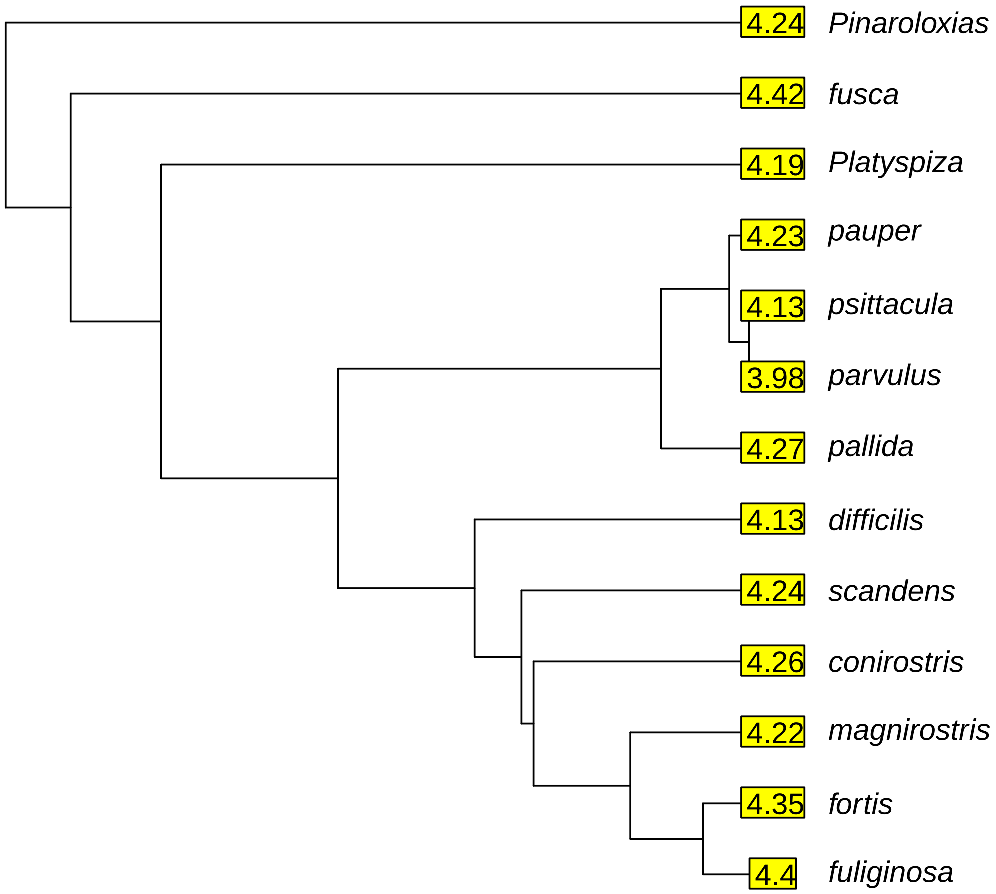
```
# pallida      4.265   3.089   2.430 2.016  1.949
# fusca        3.975   2.937   2.052 1.191  1.401
# parvulus     4.132   2.973   1.974 1.874  1.813
# pauper       4.232   3.036   2.187 2.073  1.962
# Pinaroloxias 4.189   2.980   2.311 1.548  1.630
# Platyspiza   4.420   3.271   2.331 2.347  2.282
# psittacula   4.235   3.049   2.260 2.230  2.074
```

Data are available for five different characters for all the taxa in the tree. We'll look at two of these characters, wing length and tarsus length. Extract these columns as their own variables to simplify later commands:

```
wingL <- geodata$wingL
tarsusL <- geodata$tarsusL
names(wingL) <- row.names(geodata)
names(tarsusL) <- row.names(geodata)
```

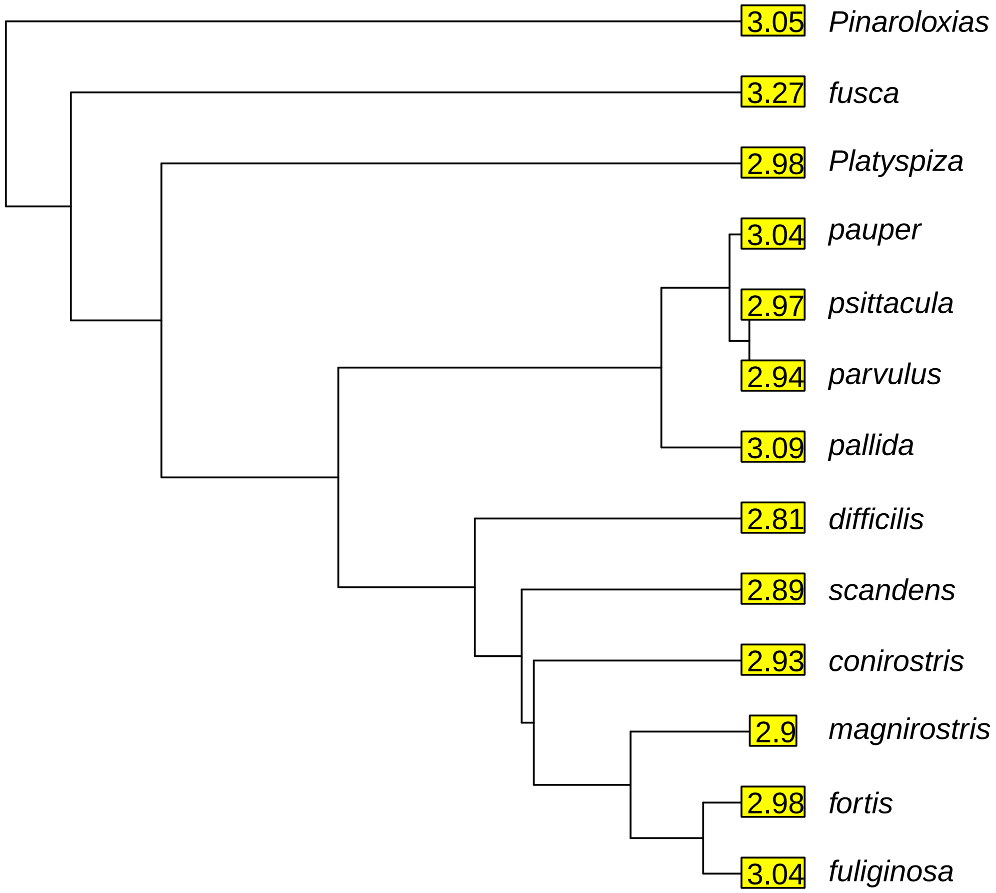
We can now plot these data directly onto the tips of the tree:

```
plot(geotree, label.offset = 0.04)
tiplabels(round(wingL, 2))
```



```
plot(geotree, label.offset = 0.04)
tiplabels(round(tarsusL, 2))
```

Independent contrasts



4 Calculating independent contrasts

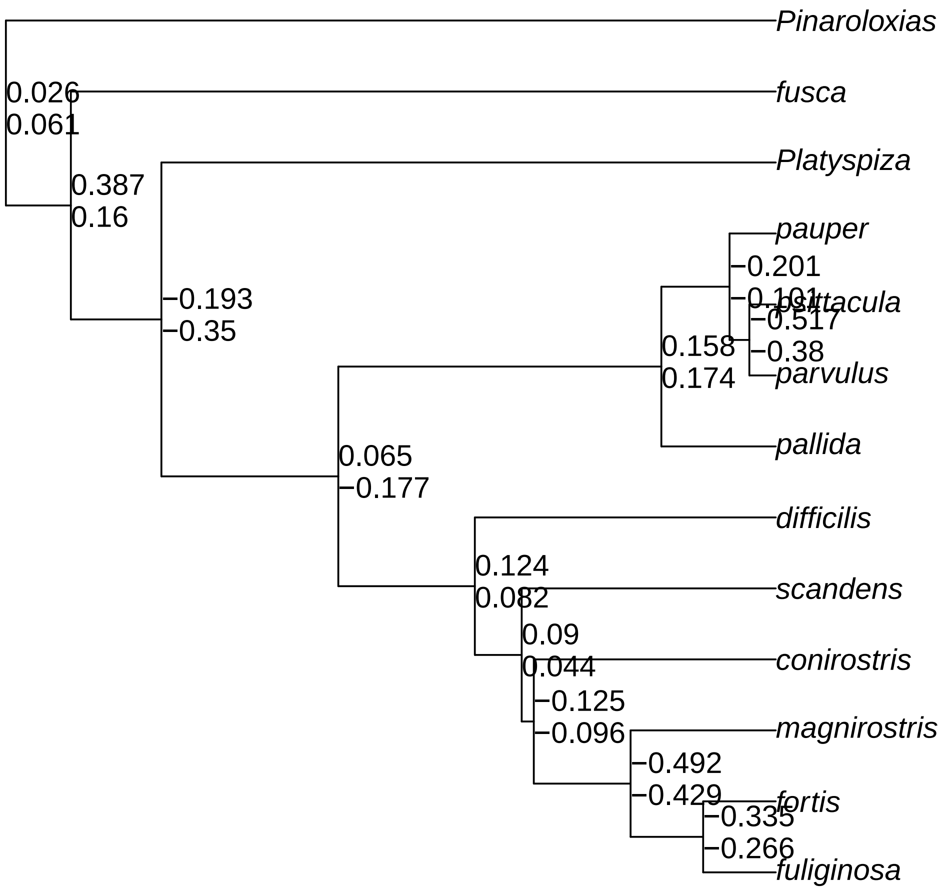
Calcualte the independent contrasts for each of the variables:

```
ContrastwingL <- pic(wingL, geotree)
ContrasttarsusL <- pic(tarsusL, geotree)
```

And plot them on the tree:

Independent contrasts

```
plot(geotree)
nodelabels(round(ContrastwingL, 3), adj = c(0, -0.5), frame = "n")
nodelabels(round(ContrasttarsusL, 3), adj = c(0, 1), frame = "n")
```

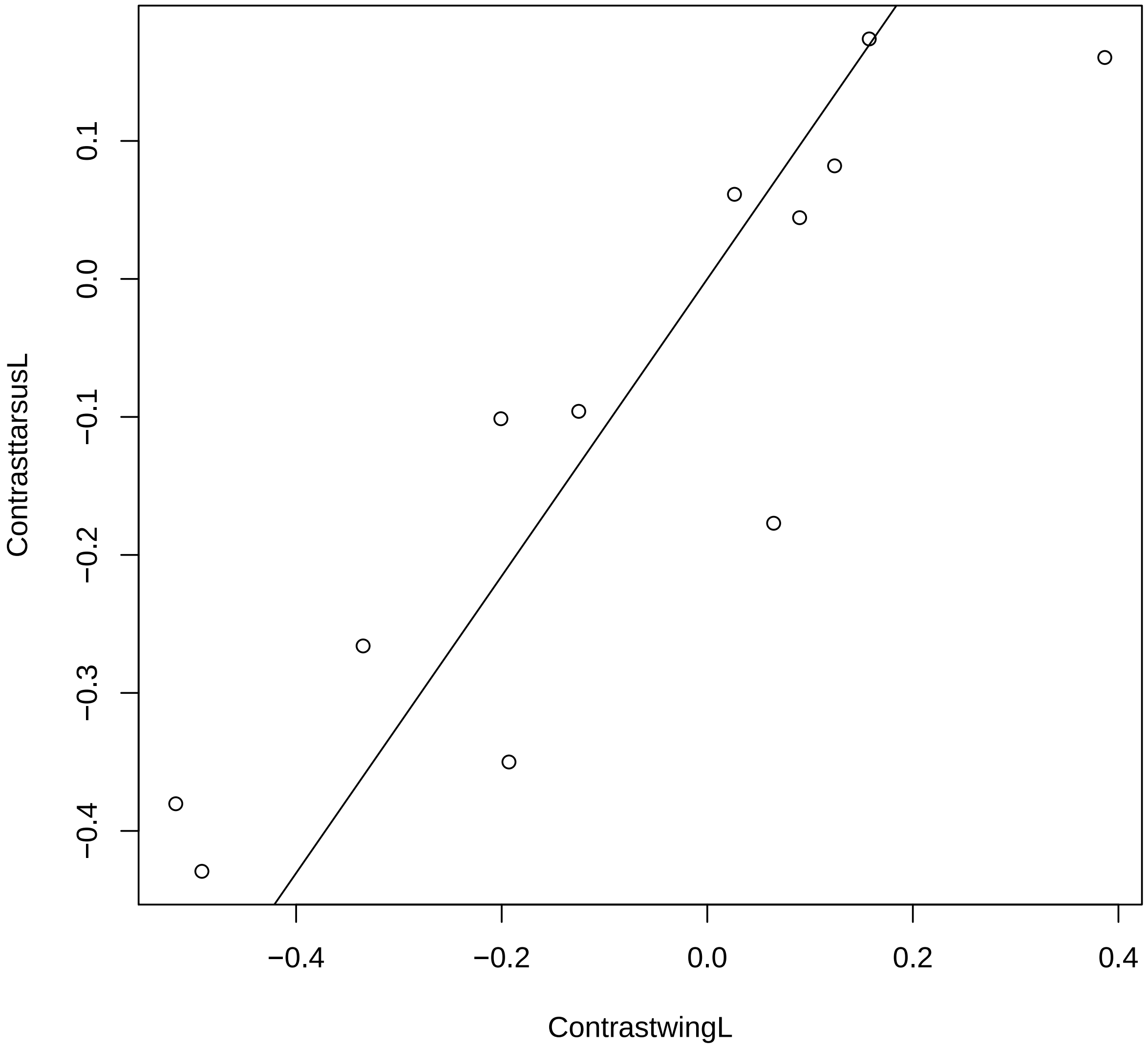


5 Investigating the relationships between the variables

```
RegressTarsusWing <- lm(ContrastwingL ~ ContrasttarsusL - 1)
summary.lm(RegressTarsusWing)
```

```
#
# Call:
# lm(formula = ContrastwingL ~ ContrasttarsusL - 1)
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -0.1077 -0.0418 -0.0257  0.0775  0.2551
#
# Coefficients:
#                 Estimate Std. Error t value Pr(>|t|)
# ContrasttarsusL    1.076      0.157    6.86  2.7e-05 ***
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 0.126 on 11 degrees of freedom
# Multiple R-squared: 0.81,Adjusted R-squared: 0.793
# F-statistic:   47 on 1 and 11 DF,  p-value: 2.74e-05
```

```
plot(ContrastwingL, ContrasttarsusL)
abline(RegressTarsusWing)
```



6 What if we didn't consider the phylogeny?

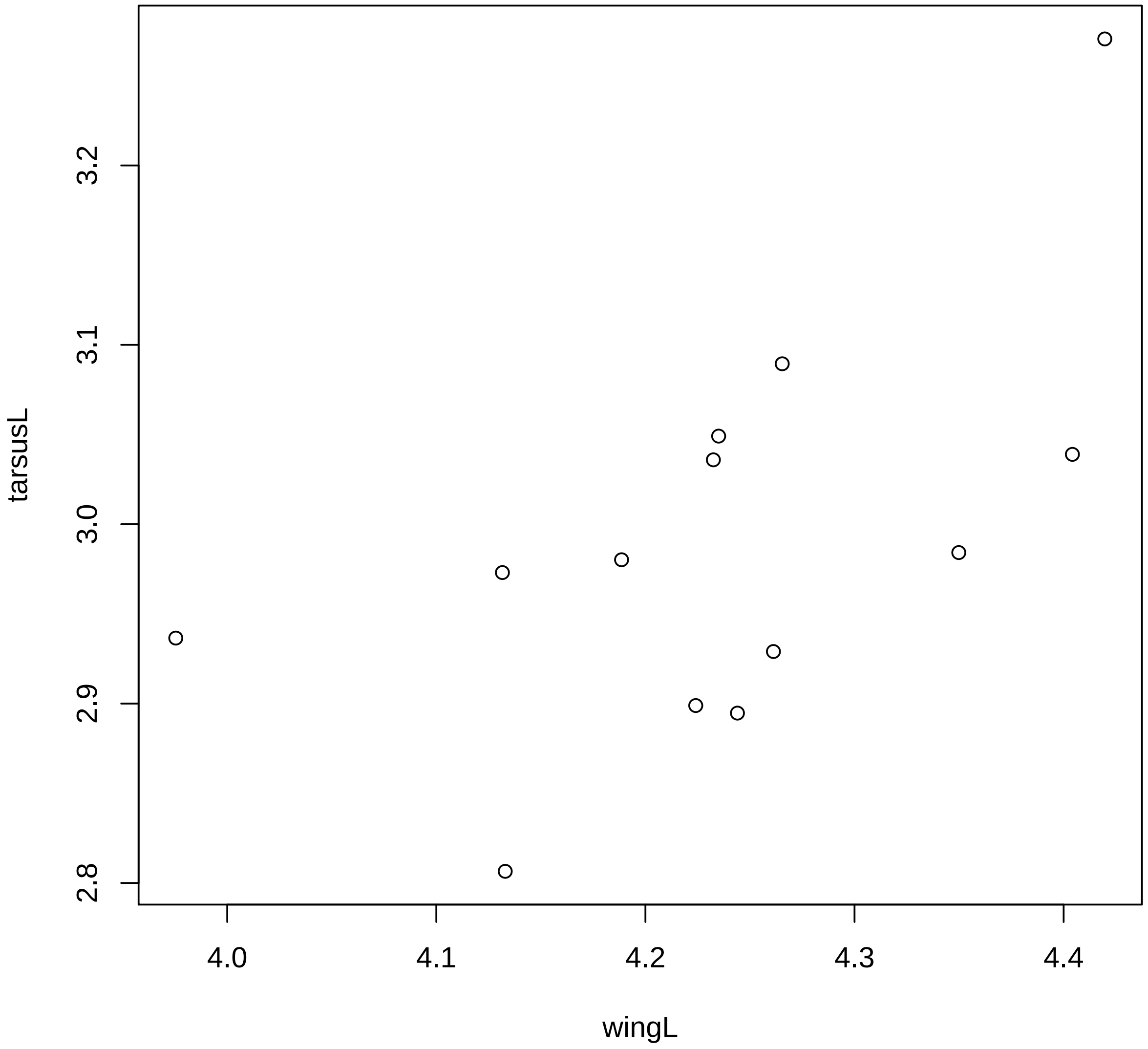
```
RegressTarsusWingNaive <- lm(wingL ~ tarsusL)
summary.lm(RegressTarsusWingNaive)

#
# Call:
# lm(formula = wingL ~ tarsusL)
#
```

```
# Residuals:
#     Min      1Q  Median      3Q     Max
# -0.2264 -0.0365  0.0112  0.0640  0.1390
#
# Coefficients:
#             Estimate Std. Error t value Pr(>|t|)
# (Intercept)    2.385      0.758    3.14   0.0093 **
# tarsusL        0.619      0.253    2.44   0.0327 *
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 0.0996 on 11 degrees of freedom
# Multiple R-squared: 0.352,Adjusted R-squared: 0.293
# F-statistic: 5.96 on 1 and 11 DF,  p-value: 0.0327
```

```
plot(wingL, tarsusL)
```

Independent contrasts



7 Understanding how the contrasts are calculated

In the above code, the function pic() took care of all the steps needed to calculate independent contrasts. In this section, we walk through these steps one by one to illustrate each of them.

We'll use a smaller tree and dataset than above. The dataset is the example dataset that is provided with the pic() function.

The following commands load the data and tree:

```
t <- "((((Homo:0.21,Pongo:0.21):0.28,Macaca:0.49):0.13,Ateles:0.62):0.38,Galago:1.00);"
tree.primates <- read.tree(text = t)

X <- c(4.09434, 3.61092, 2.37024, 2.02815, -1.46968)
Y <- c(4.74493, 3.3322, 3.3673, 2.89037, 2.30259)
names(X) <- names(Y) <- c("Homo", "Pongo", "Macaca", "Ateles", "Galago")
```

7.1 Phylogenetic trees in R

Before we get into the calculations themselves, let's lift up the hood on how the phylogenetic tree is stored. The phy tree object that is implemented in the ape library holds various data about these nodes and the edges (ie, braches) that connect them.

Take a look at the structure of our tree object.

```
str(tree.primates)

# List of 4
#  $ edge       : int [1:8, 1:2] 6 7 8 9 9 8 7 6 7 8 ...
#  $ Nnode      : int 4
#  $ tip.label  : chr [1:5] "Homo" "Pongo" "Macaca" "Ateles" ...
#  $ edge.length: num [1:8] 0.38 0.13 0.28 0.21 0.21 0.49 0.62 1
#  - attr(*, "class")= chr "phylo"
#  - attr(*, "order")= chr "cladewise"

tree.primates$edge

#      [,1] [,2]
# [1,]    6    7
# [2,]    7    8
# [3,]    8    9
# [4,]    9    1
# [5,]    9    2
# [6,]    8    3
# [7,]    7    4
# [8,]    6    5

tree.primates$edge.length

# [1] 0.38 0.13 0.28 0.21 0.21 0.49 0.62 1.00

tree.primates$Nnode

# [1] 4

tree.primates$tip.label

# [1] "Homo"   "Pongo"  "Macaca" "Ateles" "Galago"
```

The edge matrix describes how the edges connect the the nodes. It has one row per edge, and two columns. The first column gives the starting node of the edge, and the second column gives the ending node. The edge number is just the row number.

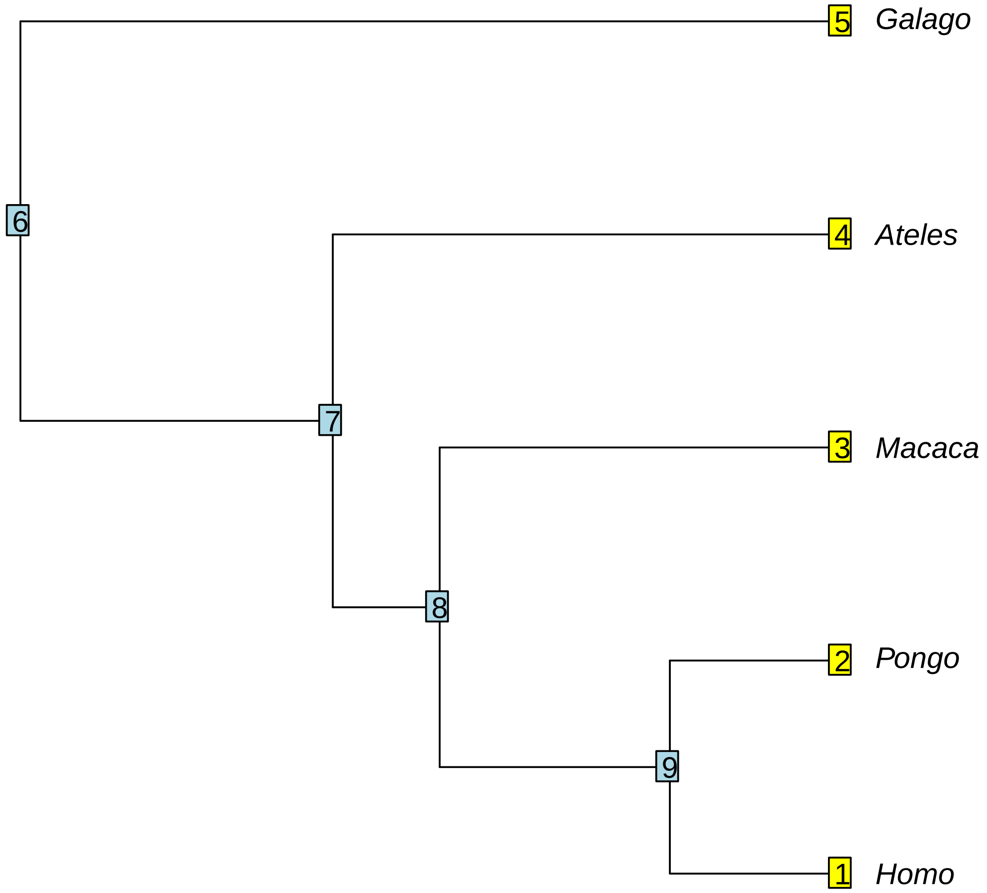
The edge.length vector contains the lengths of each of the edges.

Nnode simply holds the number of internal nodes. tip.label gives the names of the external nodes (tips).

The primary take-home message is that each node has a number that refers to it. The observed data correspond to external nodes. Ancestral reconstructions and independent contrasts will correspond to internal nodes. In addition, each edge has a number that refers to it. We can use this number to look up the edge length.

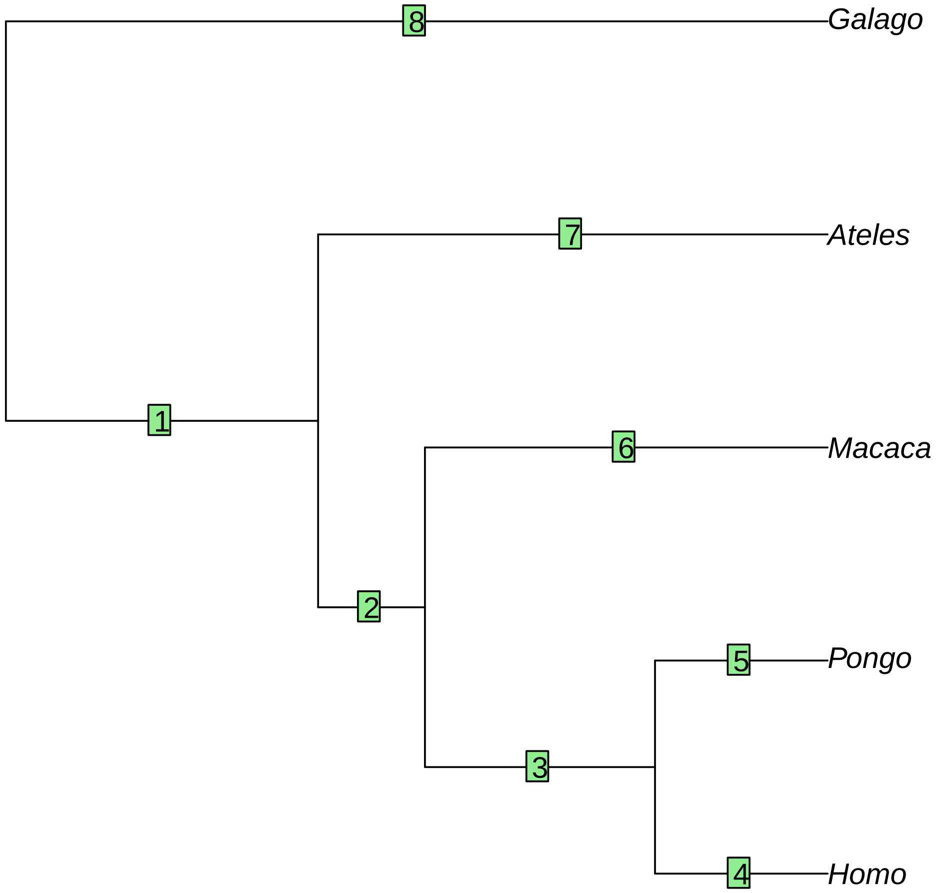
You can plot the node numbers right onto the tree. ape handles tips and internal nodes a bit differently, so we will label them on the same tree with two separate commands:

```r
plot(tree.primates, label.offset = 0.04)
tiplabels(1:5)
nodelabels(6:9)
```



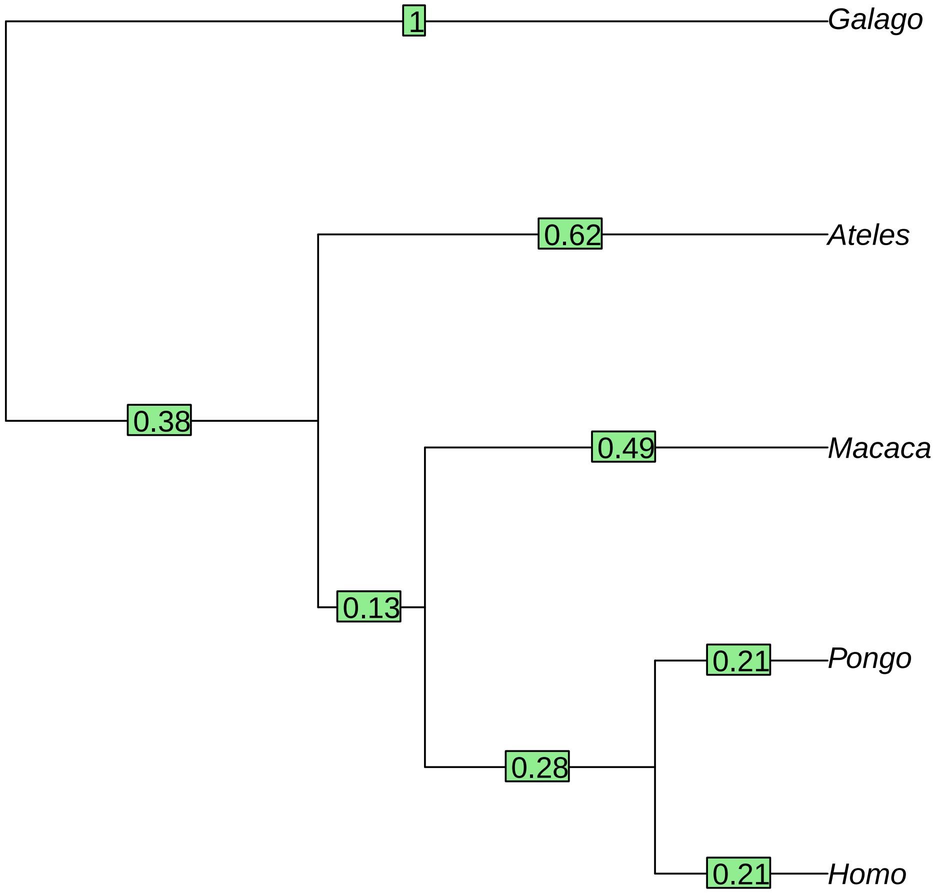
Similarly, you can plot the edge labels right onto the tree:

```
plot(tree.primates)
edgelabels(1:8)
```



And the branch lengths:

```
plot(tree.primates)
edgelabels(round(tree.primates$edge.length, 2))
```



7.2 Adjusting the branch lengths

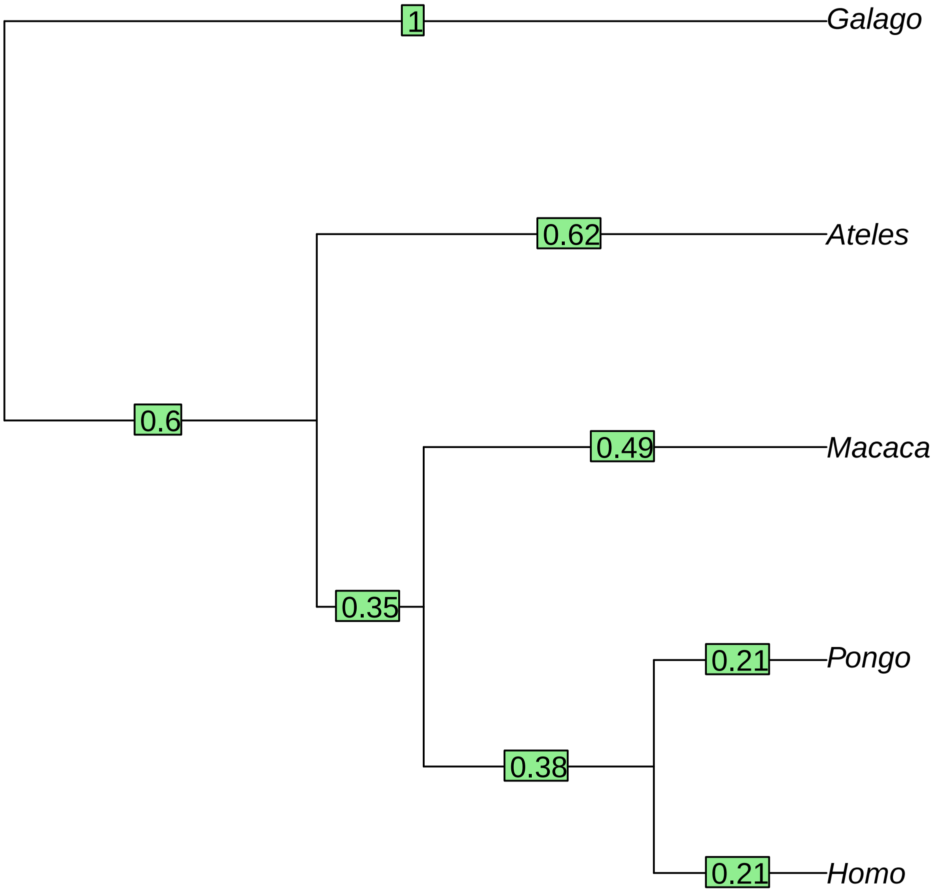
The lengths of internal branches must be adjusted to reflect the increased variance associated with the inference of ancestral character states. Below, we create a copy of all the branch lengths and then correct the lengths of the three internal branches.

```
v <- tree.primates$edge.length

v[3] <- v[3] + (v[4] * v[5])/(v[4] + v[5])
v[2] <- v[2] + (v[3] * v[6])/(v[3] + v[6])
```

```
v[1] <- v[1] + (v[7] * v[2])/(v[7] + v[2])

plot(tree.primates)
edgelabels(round(a, 2))
```



7.3 Preparing the character data

The internal character states are calculated as weighted averages of the states at the descendant nodes. This calculation is described by Equation 3 of Felsenstein (1985). Since the ancestral character state of the root node isn't used in any contrast calculations, we won't worry about it.

```
aX.9 <- (X[1] * (1/v[4]) + X[2] * (1/v[5]))/((1/v[4]) + (1/v[5]))
aX.8 <- (X[3] * (1/v[6]) + aX.9 * (1/v[3]))/((1/v[3]) + (1/v[6]))
aX.7 <- (X[4] * (1/v[7]) + aX.8 * (1/v[2]))/((1/v[7]) + (1/v[2]))

names(aX.9) <- NULL
names(aX.8) <- NULL
names(aX.7) <- NULL

aX.9

# [1] 3.853

aX.8

# [1] 3.2

aX.7

# [1] 2.781
```

A slightly different but equivalent formulation of these calculations is presented for the calculations of X_i in Table 1 of Felsenstein (1985). According to this alternative formulation, the ancestral character states for nodes 9 and 8 would be calculated as:

```
aX.9.new <- (X[1] * v[5] + X[2] * v[4])/(v[4] + v[5])
aX.8.new <- (X[3] * v[3] + aX.9 * v[6])/(v[3] + v[6])

names(aX.9.new) <- NULL
names(aX.8.new) <- NULL

aX.9.new

# [1] 3.853

aX.8.new

# [1] 3.2
```

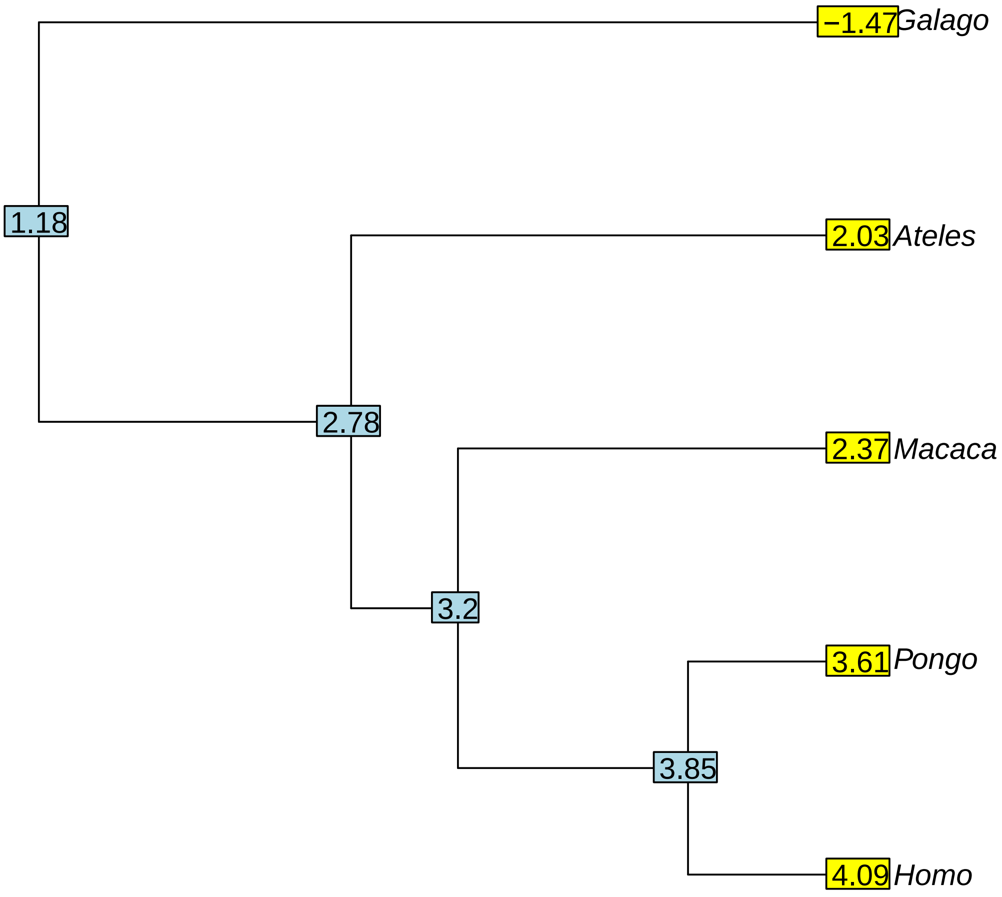
We can also calculate the ancestral character states with the ape function ace:

```
aX <- ace(X, tree.primates, method = "pic", scaled = FALSE)$ace
aX

#     6     7     8     9
# 1.184 2.781 3.200 3.853
```

Note that the values for the states at nodes 7-9 are the same as those we calculated above. Now, plot the observed values and inferred ancestral character states:

```
plot(tree.primates, label.offset = 0.04)
tiplabels(round(X, 2))
nodelabels(round(aX, 2))
```



For later convenience, combine all these state values into a single vector:

```
nX <- c(X, aX)
names(nX) <- 1:9
nX

#      1      2      3      4      5      6      7      8      9
#  4.094  3.611  2.370  2.028 -1.470  1.184  2.781  3.200  3.853
```

7.4 Calculating the contrasts

From Felsenstein (1985), the contrast at an internal node is the difference in values at the descendent nodes:

```
contrasts <- c(nX[7] - nX[5], nX[8] - nX[4], nX[9] - nX[3], nX[1] - nX[2])
names(contrasts) <- 6:9
contrasts

#      6      7      8      9
# 4.2505 1.1722 1.4824 0.4834
```

That's it. The independent contrasts are just a series of subtractions.

Now, compare the values we calculated to what we get with the pic function:

```
pic.X.ns <- pic(X, tree.primates, scaled = FALSE, var.contrasts = TRUE)
pic.X.ns

#   contrasts variance
# 6    4.2505   1.6019
# 7    1.1722   0.9656
# 8    1.4824   0.8750
# 9    0.4834   0.4200
```

The values are the same as what we got above.

For most downstream calculations, we want to scale the contrasts to normalize the expected differences according to branch lengths. I turned off this scaling above so that we could directly compare the pic() results to our own results.

The branch length is proportional to the variance. To normalize, we'll divide by the standard deviation. The square root of the variance of each contrast is the standard deviation for that contrast:

```
pic.X.ns[, 1]/sqrt(pic.X.ns[, 2])

#      6      7      8      9
# 3.3583 1.1929 1.5847 0.7459
```

These scaled vales are the same as what we get when we have pic() do the scaling for us:

```
pic(X, tree.primates, scaled = TRUE, var.contrasts = TRUE)

#   contrasts variance
# 6    3.3583   1.6019
# 7    1.1929   0.9656
# 8    1.5847   0.8750
# 9    0.7459   0.4200
```

8 How this document was made

This document is a computable data report compiled directly from the data. To recreate this file from the data, you will need to install:

- R (http://www.r-project.org). This document was generated with version 2.15.2.
- The R package knitr (http://yihui.name/knitr/), which can be installed from within R. This document was generated with version 0.9.
- pdflatex, which comes with LaTeX distributions (http://www.latex-project.org/ftp.html). This document was generated with version 3.1415926-2.4-1.40.13.

From within the knitr directory, launch R and run:

```
library(knitr)
knit("independent_contrasts.Rnw")
quit()
```

This will generate a new tex file. To compile this tex file into a pdf file, run the following at the shell command line:

```
pdflatex independent_contrasts.tex
```

In addition to recreating this document as-is, you can directly edit and add to the analyses in the .Rnw source file. You can also copy the R source code from the .Rnw file.

References

Felsenstein, J. 1985. Phylogenies and the Comparative Method. American Naturalist 125:1–15.