 # Continuous Trait R Tutorial

 # D. Luke Mahler – March 16, 2013; revised September 29, 2013

 # Summary

In this tutorial, we will conduct several exercises that employ phylogenetic comparative methods for continuous trait evolution. We'll start by loading in a phylogenetic tree and data for two traits into R, and using several tools to visualize our data, measure phylogenetic signal, and estimate ancestral states, assuming a model of Brownian motion. We'll be using published data for *Anolis* lizards, which are a well-known adaptive radiation.

We'll then conduct a "phylogenetically correct" statistical analysis, in this case a phylogenetic regression. We'll test whether perch diameter is an important predictor of relative limb length in anoles, once phylogeny is accounted for.

Third, we'll fit and formally compare several simple models for the evolutionary process that gave rise to the extant diversity of relative limb lengths in Greater Antillean *Anolis*.

 # Tutorial

 # Data Input and Preparation

#To get started, open up R, set your working directory, load the following libraries, and read in the files I gave you for *Anolis* ecomorphological and phylogenetic data. It's worth mentioning that we'll rely quite a bit on one of the most widely used phylogenetic comparative methods packages – `ape` – but we don't need to load it independently because it's loaded automatically with some of the packages below.

```
setwd("PASTE THE PATH OF YOUR FOLDER HERE")
require(geiger)
require(phytools)
require(devtools)
require(nlme)
require(qpcR)
require(reshape)
require(calibrate)
```

#Let's first import the anole phylogeny (`GA_AnolisMCC.tre`) into R.

```
GA_Anolis<-read.tree("GA_AnolisMCC.tre")
```

#How's the tree look? Type the following to plot it (note that I've decreased the font size using the parameter "`cex`"):

```
plot(GA_Anolis,cex=0.5)
```

Read in the ecology and morphology data set. The morphology data are relative limb length variables from a phylogenetic PCA conducted in Mahler et al. (2010). This data set also includes ecological data - the logarithm of the average diameters of perches used by anole species observed in the field. These data are from Glossip and Losos (1997).

```
ecomorphol <- read.csv("anole_limbs_ecol.csv",row.names=1)
```

Next, we need to make sure our data sets match (i.e., the tips of our tree need to correspond to the rows in our data frame). We'll use the base function `setdiff`, to figure out which species are in the tree but not our data set, and which are in the data set, but not the tree. Then we'll prune these species out.
Species names for phylogenetic trees in "`ape`" are stored in a vector that's part of the "`phylo`" object. We can see this vector by typing "`yourtreenamehere$tip.label`" (for us, this is "`pruned$tip.label`"). For our data object, the species names are stored as the row names.

```
TreeOnly <- setdiff(GA_Anolis$tip.label,rownames(ecomorphol))
TreeOnly # Enter the name of the object we just created to see what's in it.

DataOnly <- setdiff(rownames(ecomorphol), GA_Anolis$tip.label)
DataOnly # In our case, this should be empty.
```

In our case, everything we measured is also in the tree, but we apparently didn't measure everything in the tree. Because we have data for fewer taxa than we have in our phylogeny, we'll need to prune our tree to match it before proceeding further.
We'll prune the tree using `drop.tip`. We need to give it our tree, and a list of species to prune. We'll use the `TreeOnly` list of species names we just made to prune these species from the tree.

```
pruned <- drop.tip(GA_Anolis,TreeOnly)
```

If we had rows in our data set that weren't in the tree (i.e., if `DataOnly` contained species names), we could delete them from our data set by uncommenting the following command. What we're doing is creating a new reduced data object from `ecomorphol`. This one is made by copying the original `ecomorphol` object, minus the rows for which the `rownames` match this list of species that are in the data set, but not the tree.
Note that you don't want to use this particular command if `DataOnly` is empty, as it will delete your data. I've illustrated this command though, as this type of operation is useful for dropping rows from dataframes or matrices by row name.

```
# ecomorph2 <- ecomorphol[-match(DataOnly,rownames(ecomorphol)),]
```

Before we begin, there are a couple more things to do. First, let's sort our trait data to match the order of the tips in the tree. Many phylogenetic comparative method functions in R assume that data are arranged in an order that matches the storage of the tips of the phylogenetic tree at hand. We'll sort our morphology and ecology data this way. When running analyses in R, you should always be aware of the order of the elements in your data objects. Here's how to sort our data to match the species order from our tree:

```
ecomorphol<-ecomorphol[pruned$tip.label,]
```

Then let's make new objects with the variables of interest. The main reason we're doing this is so our script looks less cluttered when we move to fitting models and plotting. Here we'll make vectors corresponding to our limb length and perch diameter variables. Once we make each of these, we'll also name the vector elements using the species names contained in the row names in our dataframe.

```
limbs <- ecomorphol[,1] # Make a vector of limb lengths.
names(limbs) <- row.names(ecomorphol) # Name the vector elements.
perch_diameter <- ecomorphol[,2]
names(perch_diameter) <- row.names(ecomorphol)
```

 # Data Exploration

As with any analysis, the first thing we want to do is to is explore our data. This is important for a lot of reasons – it helps us catch inputting errors, visually identify interesting relationships, and interpret the results of subsequent statistical analyses.

We'll begin by plotting our variables at the tips of our tree. This way we can get a sense for the phylogenetic distribution of the trait – is a particular trait phylogenetically conserved, or labile? Does a particularly extreme trait value appear to have evolved once, or many times? It's not too difficult to plot continuous traits at the tips of your phylogeny in R. We'll use a simple script to do it. (Note that there are lots of ways to explore continuous data in R that are worth trying out. For example, the package `picante` has a function, `color.plot.phylo`, that serves much the same purpose as what we'll try below.

The first thing we'll do is rescale our variables for plotting. The following command rescales our limb measurements from 0 to 1, with the shortest-limbed species having a value of 0, and the longest having a value of 1. Ditto for perch diameter. We're not going to analyze these variables - we just want something we can plot on the tree to illustrate the variation in each trait.

```
      limbs.lines <- (limbs-min(limbs)) / (max(limbs)-min(limbs))
      perch_diameter.lines <- (perch_diameter-min(perch_diameter)) /
(max(perch_diameter)-min(perch_diameter))
```

Now let's plot the tree. Note that we're telling it to plot within a subsection of the plotting window. By setting the "`x.lim`" parameter to 3, we're telling it to plot with an upper x-axis limit of 3.

```
      plot(pruned,x.lim=3,cex=.5,font=4,label.offset=.025,edge.width=3)
```

Then we'll plot line segments corresponding to each trait. The plotting parameters in here are very specific to this data set, and should be viewed as a template. The great thing is that you can change any of them yourself – you'll need to play around with them (e.g., changing the starting positions of the segments, or their scaling) to get your own data plotted.

```
      nspecies <- length(pruned$tip.label)

      segments(rep(1.5,nspecies),1:nspecies,rep(1.5,nspecies)+(0.3*perc
h_diameter.lines),1:nspecies,lwd=3)
      mtext("relative perch diameter", at = 1.6, side = 1, line = 0,
cex=.5,font=2)

      segments(rep(2,nspecies),1:nspecies,rep(2,nspecies)+(0.3*limbs.li
nes),1:nspecies,lwd=3)
      mtext("relative limb length", at = 2.2, side = 1, line = 0,
cex=.5,font=2)
```

What can you tell about our two traits from looking at this plot? Do the traits appear to show phylogenetic signal?

There are other steps we can take to answer this question.
Let's actually calculate phylogenetic signal for these traits. Using the `phytools` function `phylosig`, we'll calculate signal using two measures: Pagel's lambda and Blomberg's K statistic.

```
      limbs.lambda <- phylosig(pruned, limbs, method="lambda")
      limbs.K <- phylosig(pruned, limbs, method="K", nsim=1000)
```

```
      perch_diameter.lambda <- phylosig(pruned, perch_diameter,
method="lambda")
      perch_diameter.K <- phylosig(pruned, perch_diameter, method="K",
nsim=1000)

      limbs.lambda
      limbs.K
      perch_diameter.lambda
      perch_diameter.K
```

Do these variables show phylogenetic signal? How would we interpret these measures, biologically? What do these values mean?

 # Ancestral State Estimation for Continuous Traits (plus Phylomorphospace plotting)

Commonly, researchers may be interested not just in what the present species values are for a trait, but what trait values the ancestral taxa within a clade had. There are many conditions under which it's very difficult to estimate ancestral states with any degree of certainty, and techniques for estimating ancestral states rely on assumptions that are commonly violated for empirical data sets. Thus, anyone interested in ancestor reconstruction should become familiar with the literature on this topic. Nonetheless, there are many situations under which such reconstructions may be useful (especially for data exploration). With the `phytools` function "`fastAnc`", we'll use maximum likelihood to estimate ancestral states for our two traits, and calculate confidence intervals for these estimates.

```
      limbs.anc <- fastAnc(pruned,limbs,CI=TRUE)
      perch_diameter.anc <- fastAnc(pruned,perch_diameter,CI=TRUE)
```

Confidence intervals for ancestral state reconstruction can be quite large, which is often the case for deep nodes.

Another way to visualize a continuous trait ancestral reconstruction is to use the `phytools` function "`contMap`".

```
      parsettings <- par(no.readonly=TRUE) # Record plotting parameters
      par(mfrow=c(1,2)) # Set plotting parameters to make 2 panel plot
      contMap(pruned,limbs,res=30,fsize=c(1,0.6))
      contMap(pruned,perch_diameter,res=30,fsize=c(1,0.6))
      par(parsettings) # Reset to original plotting settings
```

One way of visualizing the trajectory of trait evolution during diversification, assuming Brownian motion, is to plot a phylomorphospace. A phylomorphospace is a (usually) 2D plot of species values for two traits, where the species are linked by the phylogeny. The coordinates of the nodes of the phylogeny correspond to their ancestral state reconstruction values for each trait.

Before we plot a phylomorphospace, let's just plot a "morphospace." That term is a stretch here, because we're plotting limb length and perch diameter variables, only one of which is a morphological trait.

```
      plot(perch_diameter,limbs,xlim=c(-3,5),ylim=c(-10,5),xlab="log
      Perch Diameter",ylab="Relative Limb Length",pch=19,col="red")
```

Now let's plot the phylomorphospace:

```
      phylomorphospace(tree=pruned, X=cbind(perch_diameter, limbs),
label="horizontal", xlim=c(-3,5),ylim=c(-8.5,4.5), xlab="Perch
Diameter", ylab="Relative Limb Length")
```

In this phylomorphospace, the ancestral state estimates are small dots, and extant species values are larger dots.

For fun (and clarity), let's make the tip colors red, but keep the ancestor (internal node) colors black. Phylomorphospaces can be very messy, but are occasionally useful, as they illustrate ancestor-descendant trajectories of trait evolution – or at least our estimates of these trajectories assuming a Brownian motion model of evolution.

```
      tip.cols <- rep("red",length(pruned$tip.label))
      node.cols <- rep("black",pruned$Nnode)
      cols <- c(tip.cols,node.cols)
      names(cols) <- 1:length(cols)
      phylomorphospace(pruned, cbind(perch_diameter,limbs),
label="horizontal", xlim=c(-3,5),ylim=c(-8.5,4.5), xlab="Perch
Diameter",ylab="Relative Limb Length",control=list(col.node=cols))
```

 # Testing for an ecomorphological relationship in anoles using phylogenetic regression

Greater Antillean anoles have been the subject of many ecomorphological studies. Studies of organismal performance, behavior, and natural selection in the wild all suggest that there is an adaptive relationship between relative limb length and mean perch diameter in anole species. We'll test this adaptive hypothesis using the comparative method. We'll first test whether perch diameter is a significant predictor of anole limb length when phylogeny is accounted for using a regression of phylogenetically independent contrasts. We'll then compare this approach to the phylogenetic generalized least squares approach.
First, let's replot our variables:

```
      plot(perch_diameter,limbs,xlim=c(-3,5),ylim=c(-10,5),xlab="log
      Perch Diameter",ylab="Relative Limb Length",pch=19,col="red")
```

Now that we have a sense for the relationship between limb length and perch diameter, let's test the hypothesis that limb length is a function of perch diameter using a standard non-phylogenetic regression. We'll use the basic stats function "`lm`" to fit a linear model.
The notation here is pretty simple - since we've already named our independent and dependent variables, we simply put a formula into the "`lm`" function. We want it to look like: "dependent_variable ~ independent_variable". We're interested in how perch diameter has shaped limb evolution in *Anolis*, so we'll fit the following model:

```
      ecomorph.reg <- lm(limbs~perch_diameter)
```

For linear models in R, the command "`summary`" will return a table of standard statistical output.

```
      summary(ecomorph.reg)
```

We can add the regression line to the plot we made earlier with the following command.

```
      abline(ecomorph.reg)
```

Note that we can always plot a line using `abline()` by entering the intercept and slope, separated by a comma. The function will find these automatically if we give it regression output (i.e.,

ecomorph.reg), but we could plot it manually if we wanted. For example, to plot a dotted line indicating unity, type abline(0,1,lty="dotted").

So it looks like there's a highly significant relationship whereby perch height explains about a third of the variance observed.
But this could simply a phylogenetic artifact if, say, all the high-perching short-limbed species were close relatives.
Let's check this using phylogenetic comparative methods!

First, we'll calculate the phylogenetically independent contrasts for each variable, and perform a regression on these.
We'll use the ape function "pic", which takes a variable and a phylogenetic tree as input. As usual, both must either be labeled, or the variables must be ordered in the same order as the tip labels of our phylogenetic tree. We were careful to label and sort our variables earlier, so we should be OK.

```
pic.perch_diameter<-pic(perch_diameter, pruned)
pic.limbs<-pic(limbs, pruned)
```

Now that we have our contrast values, let's take a look at them. Is there an obvious relationship? Keep in mind that these points no longer correspond to species values for the two trait axes. They reflect the evolutionary differences between the immediate descendants of each node in the phylogeny.

```
plot(pic.perch_diameter,pic.limbs,xlab="log Perch Diameter
Contrast", ylab="Relative Limb Length Contrast", pch=19,
col="blue")
```

One tricky thing about independent contrasts is that any correlation between two sets of contrasts must be assessed using a regression that is "forced through the origin." That is to say, the regression line must pass through the point (0,0).

Regressing through the origin is easy in R. All we have to do is fit a linear model without an intercept term using the function "lm." We tell it not to fit an intercept by including the term "-1".

```
pic.fit <- lm(pic.limbs ~ pic.perch_diameter -1)
summary(pic.fit)
```

What's the relationship between limb length and perch diameter contrasts? Is perch height a significant predictor of limb length? Is the relationship positive or negative? Does the slope differ fundamentally from our non-phylogenetic analysis, or not really?

Now look at the multiple R-squared value to assess how much variation is explained by this predictive relationship. How does this compare to our non-phylogenetic R-squared value?

Finally, let's plot the regression line on our existing plot of contrasts:

```
abline(pic.fit)
```

##

This is great, but you'll remember from lecture that PIC is a special case of the phylogenetic generalized least squares (PGLS) model class.

Let's demonstrate this ourselves. We'll next perform a PGLS phylogenetic regression using the generalized least squares package `nlme`.
Generalized least squares is a method for estimating the fit of a linear model when the errors exhibit uneven variance (heteroscedasticity), or when the errors are correlated.
The latter is our concern - if we're looking at species, it's pretty likely we have phylogenetically correlated errors.
Functions in the package "`nlme` " allow us to specify the error structure in our model (the "correlation" term). Naturally, we'll give it a phylogenetic correlation structure, using a phylogenetic variance-covariance matrix estimated in the phylogenetics package "`ape`."
We'll start by assuming that the errors are perfectly phylogenetically correlated, which is the assumption if trait evolution occurs under pure Brownian motion. This is the same assumption made by the PIC algorithm.
We're going to use the package `nlme`, which we loaded at the start of the tutorial.

Let's fit our relationship, specifying that the correlation structure is pure Brownian motion. We'll use slightly different-looking notation for the formula here – this time the variable names are the column names from the dataframe, and we'll tell the function to use that dataframe (`ecomorphol`) to find the data:

```
result.pureBM<-gls(PCI_limbs~logPD, data=ecomorphol,
correlation=corBrownian(phy=pruned),method="ML")
summary(result.pureBM)
```

Take a look at the slope and its standard error. How do these compare to the estimates from our regression of limb PICs on perch diameter PICs?

We can plot this as well. This plot will have the same dimensions as our first (nonphylogenetic) regression plot because we're fitting a model to species values for two traits. This is an important distinction from our PIC regression. We're working in the "original species space" as opposed to a space defined by phylogenetically independent changes. This is one of the advantages of the PGLS approach - the plots are fairly intuitive.

```
plot(perch_diameter,limbs,xlim=c(-3,5),ylim=c(-10,5),xlab="log
Perch Diameter",ylab="Relative Limb Length",pch=19,col="green")
abline(result.pureBM,col="green")
```

How does the fit of our phylogenetic model compare to the non-phylogenetic one? We can see by plotting the non-phylogenetic regression line in a different color on the same plot.

```
abline(ecomorph.reg,col="black") # Fit from non-phylo regression
```

How does this fit of this regression look? Any thoughts on why it looks the way it does?

Of course the key feature of PGLS is flexibility in the assumptions of the model. For example, PIC assumes perfect phylogenetic signal (at least unless you first transform your tree).
PGLS, on the other hand, can estimate the phylogenetic signal present in the error variance of the model while it's actually fitting the model. This is more appropriate, especially if you're uncertain how much phylogenetic dependence is in your data.
Let's try this below. The PGLS model above assumed phylogenetic signal of lambda = 1, which is why we got the same answer as with PIC. Let's allow `nlme` to estimate phylogenetic signal (lambda) while fitting the model. You'll notice that the only difference is that we feed the function a different correlation structure for the errors, and tell it to estimate the phylogenetic signal while fitting the model. The correlation structure is called "`corPagel`" because we're fitting Pagel's "lambda" parameter, a common tree-scaling parameter that is used to estimate phylogenetic signal.

```
result.lambda<-gls(PCI_limbs~logPD, data=ecomorphol,
correlation=corPagel(value=1, phy=pruned,
fixed=FALSE),method="ML")
summary(result.lambda)
```

Here we'll want to look at the slope of our relationship, as before, but we'll also be interested in looking at the estimate of lambda. Lambda normally ranges from 0 - 1, with values near zero reflecting phylogenetic independence and values near 1 reflecting phylogenetic dependence. What is our estimate of lambda?

##
##

 # Exercise 2: *Anolis* Limb Length and Evolutionary Process

Greater Antillean anoles are widely regarded as an adaptive radiation, with anole species rapidly radiating across the Caribbean to utilize a diversity of microhabitat niches (e.g., tree trunks, rocks, twigs, grassy habitats). We might therefore predict that anole trait evolution is best modeled using an "early burst" evolutionary process in which evolution is rapid early in the radiation but slows down as niches are filled. However, this pattern is rarely observed in comparative data, so we'll want to consider other alternatives. In this exercise we'll investigate the evolutionary process responsible for generating relative limb length diversity in anoles. We'll use the R package `geiger` to compare the fit of three alternative macroevolutionary models: Morphological Drift (Brownian Motion), Adaptive Radiation (Early Burst), and Evolutionary Constraint (Ornstein Uhlenbeck).

First, let's read in the data. We'll be using a similar relative limb length data set, but this one has much better sampling: 100 out of the ~120 species from the Greater Antilles.

```
Limbs<-read.csv("Limbs.csv",row.names=1)
```

This "data frame" should have 100 rows, each corresponding to a Greater Antillean anole species. It should have a single column of limb length data – these values are the scores from the first axis of a principal component analysis of *Anolis* shape traits from Mahler et al. (2010).

Before we go further, let's sort our data frame so that the rows match the order of the taxa in the phylogenetic tree. This is important for many comparative analyses in R. Some functions will match your data to your tree using row labels and tip labels respectively, but not all will, so sorting is a good habit. *Technical Note: when you create a new object in R that has only a single row or column, R will automatically store it as a vector class object, even if you created it from a matrix or a dataframe. This stinks because you will lose all of your valuable row and column names! The handy command "`drop=FALSE`" will prevent this automatic class conversion.

```
Limbs<-Limbs[GA_Anolis$tip.label,,drop=FALSE]
```

Now that we have our data and our tree in order, we'll fit the three alternative models described above, saving the model parameters as data frames:

```
brown_limbs<-fitContinuous(GA_Anolis,Limbs)
```

```
      eb_limbs<-fitContinuous(GA_Anolis,Limbs,model="EB",
bounds=list(a=c(-3,1)))
      ou_limbs<-fitContinuous(GA_Anolis,Limbs,model="OU")
```

Type the names of any of these objects (e.g., `eb_limbs`) to look at the model parameters.
To compare these models easily, it helps to have a table. We'll make a table with the likelihood scores, AIC scores, delta AICs, and AIC weights so that all three models may be easily compared.

Let's do this by constructing an empty table (labeling rows and columns), which we'll then fill:

```
      model<-matrix(,3,4,dimnames = list(c("Brownian Motion", "Early
Burst", "Ornstein-Uhlenbeck"),c("log likelihood", "AICc", "Delta AICc",
"AICc Weights")))
```

Before we fill it, take a look to make sure it looks right:

```
      model
```

Now, let's put the likelihood scores and AICc values into the first two columns, calling them from the dataframes we made earlier:

```
      model[,1]<-c(brown_limbs$opt$lnL, eb_limbs$opt$lnL,
ou_limbs$opt$lnL)
      model[,2]<-c(brown_limbs$opt$aicc, eb_limbs$opt$aicc,
ou_limbs$opt$aicc)
```

ln likelihood scores can rarely be directly compared. This is why we use AICc scores for model comparison. But look at the likelihood and AICc scores for these models? What do you notice?

While AICc scores are useful, delta AICc scores and AICc weights can often be more intuitive ways of expressing model support. We'll calculate these using the `qpcR` function `akaike.weights`:

First, make a vector of AICc scores:

```
      aic.all<-as.matrix(model[,2])
```

Then, calculate delta AICc scores and AICc weights.

```
      scor.wts<-akaike.weights(aic.all)
```

Finally, add these to our table:

```
      model[,3]<-scor.wts$deltaAIC
      model[,4]<-scor.wts$weights
```

How does it look? Which, if any, model is best supported?

```
      model
```

This is great, but we're not done yet! Naturally, we want to dig deeper and explore the parameters of our favored model. In particular, we'll want to assess the level of certainty we have in both our model selection and our parameter estimates.
While AIC scores are a very straightforward way to compare models, a recent study (Boettiger et al. 2012, Evolution 67:2240-2251) has shown that this model selection approach may erroneously favor overly complex models for comparative data (i.e., high Type I error).

Thus, we'll next show how to more rigorously compare models using parametric bootstrap simulations. We'll also show how to quantify phylogenetic uncertainty in model parameter estimates by repeating an analysis across a distribution of trees.

############################

(A) Repeating Analyses Across a Tree Distribution to Account for Phylogenetic Uncertainty

############################

Methods that use maximum likelihood to select models of trait evolution typically assume that the phylogenetic history of the focal organisms is known without error. This is unlikely to be the case, of course. Usually we use a particular phylogeny that we like for some reason – perhaps it's the "most credible" phylogeny from a Bayesian posterior distribution according to some criterion, for example. Although many Bayesian methods account for phylogenetic uncertainty natively when estimating parameters, we'll have to do this ourselves when using the ML methods introduced above.

Perhaps the first thing we'll want to know is whether our model selection is robust to phylogenetic uncertainty. Let's read in a series of trees from a Bayesian posterior distribution (to save time, we'll only consider a 50 tree sample here).

```
trees50 <- read.tree("trees.mid.pruned.tre")
```

Next, the following script shows how to fit our four models using all 50 trees in this Bayesian distribution. Note: I've commented these commands out because they take several minutes to run (more time than we have now). What we'll do instead is read in the results of these analyses (I ran them earlier, and saved a file for each result object).

```
treesconsidered <- 50

# brown_trees <- list(length=treesconsidered)
# eb_trees <- list(length=treesconsidered)
# ou_trees <- list(length=treesconsidered)

# for(i in 1:treesconsidered){
      # brown_trees[[i]] <- fitContinuous(trees50[[i]],Limbs)
      # eb_trees[[i]] <- fitContinuous(trees50[[i]],Limbs,
model="EB",bounds=list(a=c(-5,1)))
      # ou_trees[[i]] <- fitContinuous(trees50[[i]],Limbs,
model="OU")
# }

load("brown_trees.R")
load("eb_trees.R")
load("ou_trees.R")
```

Now that we've fitted the models, let's organize the information we need to compare them. The question we want to answer is:

"Is model support consistent across all of the phylogenies in our sample?"

You'll recall we used AICc scores to compare models earlier. We'll do this again here, as our current objective is to look at phylogenetic variation in relative model support. Let's pull AICc information out for our sample of trees. First we'll make a table:

```
      aic.table <- matrix(,treesconsidered,3,dimnames=list(c(1:
treesconsidered),c("Brownian Motion", "Early Burst", "Ornstein-
Uhlenbeck")))
```

Now let's fill it with the AICc scores from our model fitting results above.

```
      for(i in 1:treesconsidered){
      aic.table[i,1] <- brown_trees[[i]]$opt$aicc
      aic.table[i,2] <- eb_trees[[i]]$opt$aicc
      aic.table[i,3] <- ou_trees[[i]]$opt$aicc
      }
```

AIC weights are a little more intuitive, so let's go a step further and make a table that displays these for the alternative models for all of our trees. This is pretty similar to what we just did. First we'll make an empty table, and then we'll fill it with AICc weights.

```
      aic.weights.table <- matrix(,treesconsidered,3,dimnames=list(c(1:
treesconsidered),c("Brownian Motion", "Early Burst", "Ornstein-
Uhlenbeck")))

      for(i in 1: treesconsidered){
      aic.weights.table[i,] <- akaike.weights(aic.table[i,])$weights
      }
```

It will be easier to compare our models if we plot the distributions of weights graphically.
Density plots can be useful for comparing distributions. We'll first generate and store kernel density estimates for our distributions. We'll then plot them all on the same plot (I'm "filling" these plots with the "`polygon`" function, and making them semi-transparent by calling "`rgb`" to color them in. The last value in an "`rgb`" command determines the degree of transparency of the plotted color.).

```
      bm.aic.weights <- density(aic.weights.table[,1])
      eb.aic.weights <- density(aic.weights.table[,2])
      ou.aic.weights <- density(aic.weights.table[,3])

      plot(bm.aic.weights,xlim=c(0,1),ylim=c(0,25),main="Model support
over distribution of phylogenies",xlab="AICc
weight",col=rgb(0,0,0,0.5))
      polygon(bm.aic.weights,col=rgb(0,0,0,0.5))
      polygon(eb.aic.weights,col=rgb(1,0,0,0.5))
      polygon(ou.aic.weights,col=rgb(0,0,1,0.5))
      legend("topright", c("BM", "EB", "OU"), lty=1, lwd=4,
col=c(rgb(0,0,0,0.5), rgb(1,0,0,0.5), rgb(0,0,1,0.5)))
```

So this tells us how phylogenetic uncertainty affects model selection, but how does it affect parameter estimation? We can quantify this uncertainty as well using the same methods. Let's focus on the Early Burst model and take a look at how phylogenetic uncertainty affects our estimate of the "slowdown" parameter.

```
      eb.alpha.table <- matrix(,treesconsidered,1,dimnames=list(c(1:
treesconsidered),c("alpha")))

      for(i in 1:treesconsidered){
            eb.alpha.table[i,1] <-  eb_trees[[i]]$opt$a
      }
```

```
      phylo.density <- density(eb.alpha.table[,1])
      plot(phylo.density,main="Phylogenetic uncertainty for the
slowdown parameter", xlab="alpha",col="darkgray", xlim=c(-3,0),
ylim=c(0,2.5))
      polygon(phylo.density,col="darkgray")
```

For the early burst model, does there seem to be much variation in our estimate of alpha?

############################

(B) Using Parametric Bootstrapping to Quantify Parameter Estimation Uncertainty

############################

Phylogenetic uncertainty is only one source of uncertainty though. There's also uncertainty associated with our estimation of model parameters. Here we'll employ a "bootstrapping" simulation technique (simulation is used to numerically assess the sampling properties of our method of estimating parameters). Fortunately for us, Carl Boettiger wrote an R package (`pmc`) that can be used to obtain bootstrapped estimates of parameter estimates for models of continuous trait evolution.
Before we get started, we need to do a little hack to make sure `pmc` works. The `pmc` package depends on an old version of `geiger`, and many of the functions in that package changed with a recent update. `pmc` hasn't been updated to use the current version of `geiger` yet. So…. we're going to enter developer mode (using `devtools`) to temporarily install the old version of `geiger` from GitHub just for this session, and then install `pmc`, so that `pmc` will work.

```
      dev_mode(on=TRUE)

      install_github("geiger", "cboettig")
      install.packages("pmc")
      require(pmc)
```

We'll use it to simulate expected distributions for the parameters of the model that provided the best fit to our limb length data: the Early Burst model (`pmc` simulates for pairs for model comparison purposes, so well also simulate under Brownian Motion). Note that we're using a relatively small number of bootstraps. This is simply to save time, and a real study should use an order of magnitude more.

First, we simulate. When we run "`pmc`", we simulate under two models (here, we're simulating 100 data sets using the empirical parameters for the Early Burst model, and then 100 data sets using the empirical Brownian Motion parameters).
What we're interested in looking at is the distribution of "slowdown parameter" values we obtain by simulating under the Early Burst model. This provides a robust way to obtain confidence intervals for this parameter.

What we'll do is simulate with 10 bootstraps, but since bootstrapping takes awhile, we'll then load in a pre-simulated data set with 100 simulations.

```
      pmc_eb_limbs <-
pmc(GA_Anolis,Limbs,modelA="EB",modelB="BM",nboot=10)
```

Here's the data set with 100 sims. By loading this, we're writing over our old "`pmc_eb_limbs`" object with a bigger one.

```
      load("alpha_parameter_bootstrap.R")
```

Now pull out the subset of `pmc` results corresponding to the "`a`" parameter.

```
      aparameter <- subset(pmc_eb_limbs[["par_dists"]],comparison ==
"AA" & parameter == "a")
```

Look at the bootstrapped distribution of the "slowdown" parameter.

```
      bootstrap.density <- density(aparameter[,1])
      plot(bootstrap.density,main="Estimation uncertainty for the
slowdown parameter", xlab="alpha",col="darkgray",xlim=c(-3,2),
ylim=c(0,1.5))
      polygon(bootstrap.density,col="darkgray")
```

We can calculate confidence intervals using this bootstrap distribution using the `cast` command from the package `reshape`.
Here's the command to return the 1-tailed 95% confidence interval.

```
      cast(aparameter, comparison ~ parameter, function(x) quantile(x,
c(0.0, 0.95)), value = c("lower", "upper"))
```

```
############################
```

(C) Using Parametric Bootstrapping to Quantify Phylogenetic and Parameter Estimation Uncertainty Simultaneously

```
############################
```

Of course what we'd like to do is quantify both phylogenetic and model uncertainty. Let's do this by creating a loop that performs our bootstrapping operation over the distribution of trees we specified earlier.
Because bootstrapping is very time intensive, I've commented out this code, and have provided results from simulations I ran overnight instead. Nonetheless, it's worth going over this part of the script.

First we'll create a matrix with no rows.
We'll then add rows to this matrix with our loop. Each time we do bootstrapping, we add a single row to our matrix for each simulation (so since we set `nboot = 100`, that's 100 rows). We simply repeat this for each phylogeny from our distribution of phylogenies - we therefore add 100 rows per phylogeny for a total of 5000 rows in this exercise. Each column in the matrix should have some parameter, and the distribution of this parameter across all of our rows should reflect the distribution of our model parameter given uncertainty in both model estimation and phylogeny. We can visualize this by plotting a frequency distribution of values in a column (i.e., parameter) of interest.

```
      # a_bootstrap_phylo <- matrix(NA,0,4)
      # for(i in 1:50){
            # Limbs<-Limbs[trees50[[i]]$tip.label,,drop=FALSE] # Don't
forget to sort your trait data for each loop so that it matches the
order in your tree.  This is CRITICAL here.
            # bootstrap <-
pmc(trees50[[i]],Limbs,modelA="EB",modelB="BM",nboot=100)
            # a_bootstrap_phylo <-
rbind(a_bootstrap_phylo,subset(bootstrap[["par_dists"]],comparison ==
"AA" & parameter == "a"))
```

```
	# }
```

Let's load up the pre-simulated results.

```
	load("alpha_phylo_bootstrap.R")
```

Now let's plot the distribution of alpha, accounting for uncertainty in parameter estimation, as well as phylogenetic uncertainty. This is a parametric bootstrap approach to obtaining a confidence distribution for a parameter of interest.

```
	bootstrap.phylo.density <- density(a_bootstrap_phylo[,1])
	plot(bootstrap.phylo.density,main="Phylogenetic and estimation
uncertainty for the slowdown parameter",
xlab="alpha",col="darkgray",xlim=c(-3,2), ylim=c(0,1.5))
	polygon(bootstrap.phylo.density,col="darkgray")
```

Let's again return the 1-tailed 95% confidence interval.

```
	cast(a_bootstrap_phylo, comparison ~ parameter, function(x)
quantile(x, c(0.0, 0.95)), value = c("lower", "upper"))
```

Let's make a plot that compares the sources of uncertainty in our slowdown estimate.

```
	par(mfrow=c(3,1))

	plot(phylo.density,main="Phylogenetic Uncertainty",
xlab="alpha",col="red",xlim=c(-3,1))
	polygon(phylo.density,col="red")

	plot(bootstrap.density,main="Estimation Uncertainty",
xlab="alpha",col="blue",xlim=c(-3,1))
	polygon(bootstrap.density,col="blue")

	plot(bootstrap.phylo.density,main="Phylogenetic and Estimation
Uncertainty", xlab="alpha",col="black",xlim=c(-3,1))
	polygon(bootstrap.phylo.density,col="black")

	par(parsettings)
```

That's it! Let's turn developer mode back off before moving on to anything else. Note that when you restart R, you'll be back to the current version of `geiger`.

```
	dev_mode(on=FALSE)
```