Phylogenetic Diversity - Traits

Z620: Quantitative Biodiversity, Indiana University

February 17, 2017

OVERVIEW

Up to this point, we have been focusing on patterns taxonomic diversity in Quantitative Biodiversity. Although taxonomic diversity is an important dimension of biodiversity, it does not consider the evolutionary history or relatedness of species. The goal of this exercise is to introduce basic concepts of phylogenetic diversity.

After completing this exercise you will know how to:

1. create phylogenetic trees
2. map traits onto phylogenetic trees
3. test for phylogenetic signal of traits on a phylogenetic tree

1) SETUP

A. Retrieve and Set Your Working Directory

```
rm(list = ls())
getwd()
setwd("~/GitHub/QB-2017/Week6-PhyloTraits")
```

B. Load Packages

There are many contributed R packages that have been developed for conducting phylogenetic analyses (http://goo.gl/DtU16j). We will rely heavily on the `ape` (http://goo.gl/nExs0), which peforms a wide range of phylogenetic analyses. We will supplement this with other packages for some specialized tasks such as reading of aligned sequences (`seqinr`), and testing for phylogenetic signal (`geiger`). We will also align sequences using the program `muscle` through the terminal. You may also want to check out the `phangorn` package (http://goo.gl/iWU6Vt), which is good for conducting maximum likelihood analyses in R. It is also worth looking into the Randomized Axelerated Maximum Likelihood (RAxML) package for generating maximum likelihood values with bootstrap support (http://sco.h-its.org/exelixis/web/software/raxml/index.html). For large numbers of taxa programs like RAxML may be too slow. Here you can use the program FastTree to generate approximately-maximum-likelihood phylogenetic trees, sacrificing exactness for speed. However, both RAxML and FastTree will have to be run in the terminal.

After the initial installation of these packages using the `install.packages()` fucntion, let's load the packages and their dependencies with the `require()` function:

```
package.list <- c('ape', 'seqinr', 'phylobase', 'adephylo', 'geiger', 'picante', 'stats', 'RColorBrewer
for (package in package.list) {
  if (!require(package, character.only=TRUE, quietly=TRUE)) {
    install.packages(package)
    library(package, character.only=TRUE)
  }
}
```

2) DESCRIPTION OF DATA

The maintenance of biodiversity is thought to be influenced by **trade-offs** among species in certain functional traits. One such trade-off involves the ability of a highly specialized species to peform exceptionally well on a particular resource. This strategy may be advantageous if the specialist lives in an environment where its preferred resource is found in a relatively high and constant supply. However, the specialist strategy may not be optimal if the species finds itself in an environment where resource conditions are more variable. Under these conditions, it may be beneficial if a species is capable of using a wider range of resources, even though there may be a fitness cost associated with being a generalist. We set out to test this hypothesized trade-off in aquatic bacteria by assessing generalist and specialist strategies on different forms of phopshorus, a resource which commonly limits growth and reproduction of organisms inhabiting freshwater ecosystems. We isolated 39 strains of bacteria from two lakes. After identifying these bacteria based on their 16S rRNA gene sequences, we measured their growth on 18 different forms of phosphorus. In this exercise, we will take a phylogenetic approach to mapping phosphorus resource use onto a phylogenetic tree while testing for specialist-generalist trade-offs.

3) SEQUENCE ALIGNMENT

After generating or retrieving sequence data, the first step of a phylogenetic analysis is to peform an alignment. This can be done with nucleotides (DNA or RNA sequences) or proteins (amino acid sequences). There are various methods for aligning sequences, but the basic premise is to use an algorithm to arrange sequences relative to one another based on conserved regions so that downstream phylogenetic analyses can be performed. In R, there are at least two packages that can be used for aligning sequences: `ape` and `muscle`. In `ape`, alignments can be done with the functions `clustal`, `muscle`, and `tcoffee`, all which have a range of options. However, these functions require that other programs be installed on your computer. Therefore, for this exericse, we are going to use the `muscle` package in R becasue it will align sequences in a "stand alone" fashion without the need for installing additional software.

A. Examining a FASTA-File

We will read in a FASTA-formatted file into the `muslce` alignment function. This is a very common text-based format that is used across platforms and disciplines. Let's take a second to look at the *p.isolates.fasta* file. Open the terminal and change directory to the data folder. You can then type "nano p.isolates.fasta" at the command line. This will open the FASTA file in nano, which is a text editing program. You can use [Ctrl+V] and [Ctrl+Y] to scroll towards the bottom and top of the file, respectively. When you are done, you can close the file by hitting [Ctrl+X]. Do not save any inadvertent changes to *p.isolates.fasta*.

B. Performing an Alignment

Now, let's align our sequences. Using the `muscle` fuction, we will read in our FASTA file and create a FASTA-aligned output file (with an .afa extension).

```
muscle -in ./data/p.isolates.fasta -out ./data/p.isolates_test.afa
```

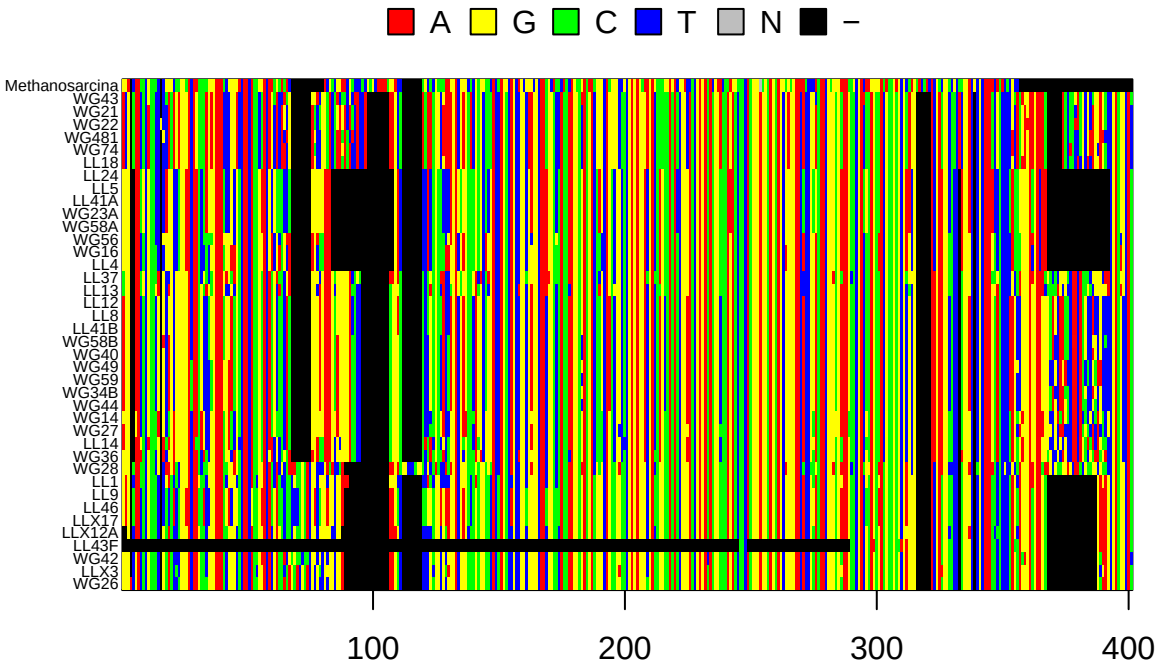
Question 1: Using nano or your favorite text editor, compare the `p.isolates.fasta` file and the `p.isolates.afa` file. Describe the differences that you observe between the files.

Answer 1:

C. Visualizing the Alignment

It is good practice to view your sequences after alignment. This may not be practical for large databases, but in our example, the alignment only contains sequences for 39 bacterial species plus a distantly related reference sequence that we will use as an **outgroup**. To visualize our alingment, first, we will read in the FASTA-aligned file using the `read.alignment` function in the `seqinr` package. Then, using the `ape` package, we will convert the alignnment file it into a DNAbin object, which is a bit-level coding scheme that allows R to store and manipulate sequence data. Last, we will visualize the alignment by color-coding nucelobases different colors, where red = adenine, yellow = guanine, green = cytosine, blue = thymidine, grey = ambiguous call, and black = alignment gaps.

```
# Read Alignment File {seqinr}
read.aln <- read.alignment(file = "./data/p.isolates.afa", format = "fasta")

# Convert Alignment File to DNAbin Object {ape}
p.DNAbin <- as.DNAbin(read.aln)

# Identify Base Pair Region of 16S rRNA Gene to Visuzlize
window <- p.DNAbin[, 100:500]

# Command to Visusalize Sequence Alignment {ape}
image.DNAbin(window, cex.lab = 0.50)
```




```
# Optional Code Adds Grid to Help Visualize Rows of Sequences
# grid(ncol(window), nrow(window), col = "lightgrey")
```

Question 2: Make some observations about the `muscle` alingmnet of the 16S rRNA gene sequences for our bacterial isolates and the outgroup, *Methanosarcina*, which belongs to the archaeal domain of life. Move along the alignment by changing the values in the `window` object.

a. Approximatley how long were our reads?

b. What regions do you think would be most appropriate for phylogenetic inference?

Answer 2a:

Answer 2b:

4) MAKING A PHYLOGENETIC TREE

Once you have aligned your sequences, the next step is to construct a phylogenetic tree. Not only is a phylogenetic tree effective for visualizing the evolutionary relationship among taxa, but as you will see later, the information that goes into a phylogenetic tree is needed for downstream analysis.

We could spend an entire semester discussing the theory and practice behind making phylogenetic trees. The objective of this handout is to quickly get you up to speed on some of the basic principles of phylognetic reconstruction. In adition, we want to convey some of the opportunities and limitations of conducting phylogenetic analyses in R. The following table provides an overview of the methods that are available for evolutionary reconstruction in R.

Tree Method	Properties
UPGMA	Hierarchical clustering technique. Progressively groups the most similiar pairs of sequences. Sometimes used as a guide tree for more sophisticated phylogenetic analyses, but prone to distorting tree topology and assumes constant rates of evolution. UPGMA can be implemented in R with the *upgma()* function in the *panghorn* package.
Neighbor Joining	Starts with an unresolved tree in a "star network". Algorithm then identifies the two most similar taxa and creates a node. This process is iteratively applied to the remaining sequences. Can serve as a starting point for assessing evolutionary models with more sophisticated methods like maximum likelihood. Neighbor joining can be implemented in the R package *ape* using the *bionj()* function.
Maximum Parsimony	A character-based, non-parametric approach that involves identifying one of many possible trees based on the minimum number of evolutionary "steps" (or mutations). No underlying evolutionary model. Maximum parsimony can be executed using the *parsimony()* function in the package *panghorn*. The function *optim.parsimony* performs tree rearrangements using nearest-neighbor interchanges (NNI) to optimize parsimony scores.
Maximum Likelihood	Parameteric approach to identifying the relative probability associated with data fitting a given model of character evolution. One of the most commonly used approaches for building trees from sequence data. Statistically sound and flexible for accomodating realistic models of sequence evolution. Maximum likelihood can be implemented in R using the *pml()* function in the *panghorn* package. Parameters can be optimized using the *optim.pml()* function and evolutionary models can be compared with the *model.tests()* function
Bayesian	Uses likelihood function and prior probabilities to create a posterior probability of a tree fitting a given model of evolution. Computationally intensive; often uses Markov chains Monte Carlo (MCMC). Unlike other methods, parameters are not constant; rather, they come from probabilistic distributions. Along with maximum likelihood, considered one of the more preferable ways of generating trees. R has some packages (e.g., *ips*) the wrap popular Bayesian software such as MrBayes

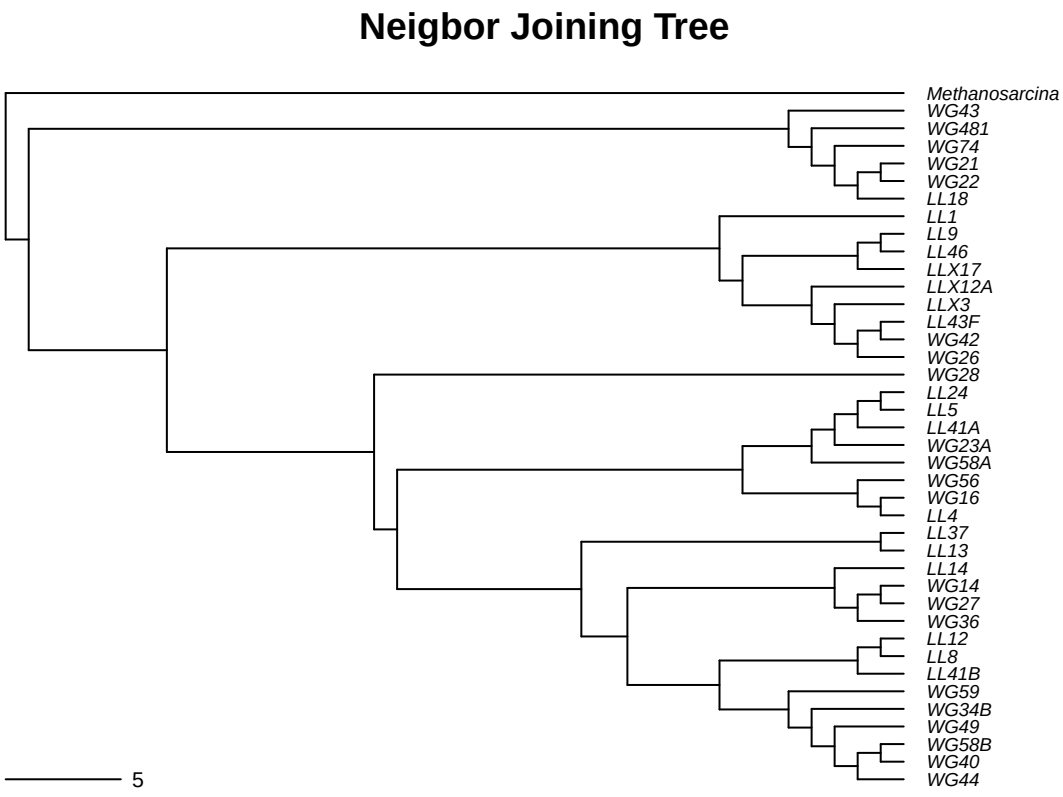
A. Neighbor Joining Trees

For the purposes of this course, we are going to focus primarily on building trees using neighbor joining methods. As you will see in the next section, neighbor joining is commonly used for making “guide tree” that can incorporate more sophisticated models of evolution. The first step in making a neighbor joining tree is to create a distance matrix. We will do this with the `dist.dna` function in the `ape` package. In the R chunk below, the “model = raw” argument means that `dist.dna` will estimate distances based on the proportion of sites that differ between pairs of sequences. The “pairwise.deletion = FALSE” argument means that an entire site (i.e., column) is deleted if there is a missing observation (i.e., a gap) for one or more of the sequences (i.e., rows).

```
# Create Distance Matrix with "raw" Model {ape}
seq.dist.raw <- dist.dna(p.DNAbin, model = "raw", pairwise.deletion = FALSE)
```

After calculating the distances between our sequences, we are now ready to make a neighor joining tree using the `bionj` function in `ape`. In the process, we will identify an outgroup sequence (*Methanosarcina*) that we will use to root the tree. At this point, we can plot our tree.

```
# Neighbor Joining Algorithm to Construct Tree, a 'phylo' Object {ape}
nj.tree <- bionj(seq.dist.raw)

# Identify Outgroup Sequence
outgroup <- match("Methanosarcina", nj.tree$tip.label)

# Root the Tree {ape}
nj.rooted <- root(nj.tree, outgroup, resolve.root = TRUE)

# Plot the Rooted Tree{ape}
par(mar = c(1,1,2,1) + 0.1)
plot.phylo(nj.rooted, main = "Neigbor Joining Tree", "phylogram", use.edge.length = FALSE,
           direction = "right", cex = 0.6, label.offset = 1)
add.scale.bar(cex = 0.7)
```



B) SUBSTITUTION MODELS OF DNA EVOLUTION

Phylogenetic analyses require assumptions about the rates of DNA evolution. Various models have been developed to estimate the rates that nucleotides change over time. These models are important for inferring relatedness of genes and taxa. In the following table, we describe a few of the more commonly used models.

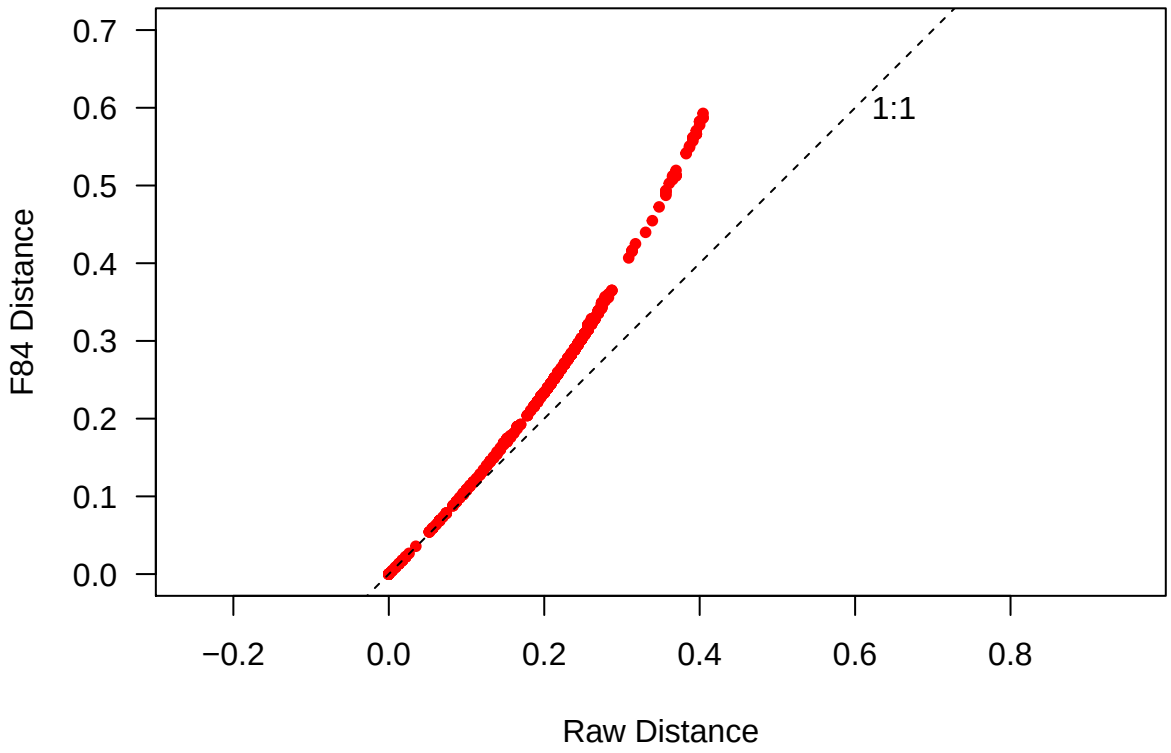
Evolutionary Model	Properties
Jukes-Cantor model (JC69)	Simplest model. Assumes all nucleotides occur at equal frequencies and that these nucleotides can mutate from one to another with equal probability.
Felsenstein model (F81)	Builds from JC69 by allowing nucleotide frequencies to vary.
Kimura model (K80)	Assumes equal frequencies of nucleotides, but recognizes that transition mutations (e.g., purine to a purine [A -> G] or pyrmidine to pyrmidine [C -> T]) occur with higher probability than tranversion mutations (e.g., purine to pyrmidine or vice-versa).
Felsenstein model (F84)	Assumes different rates of base transitions and transversions while allowing for differences in base frequencies.
Tamura model (T92)	Similar to K80 but accounts for G + C content.
General Time Reversible model (GTR)	Captures the probabilities associated with nucleotide reversions (e.g., T -> C -> T). All substitution rates are different. Does not assume equal base frequencies.

Fortunately, the `dist.dna` function that we used above for generating the neighbor joining tree can easily create distance matrices for a large number of DNA substitution models. Let's create a distance matrix among our bacterial isolates based on the Felsenstein (F84) subsitution model using the `dist.dna` function in `ape`.

```
# Create distance matrix with "F84" model {ape}
seq.dist.F84 <- dist.dna(p.DNAbin, model = "F84", pairwise.deletion = FALSE)
```

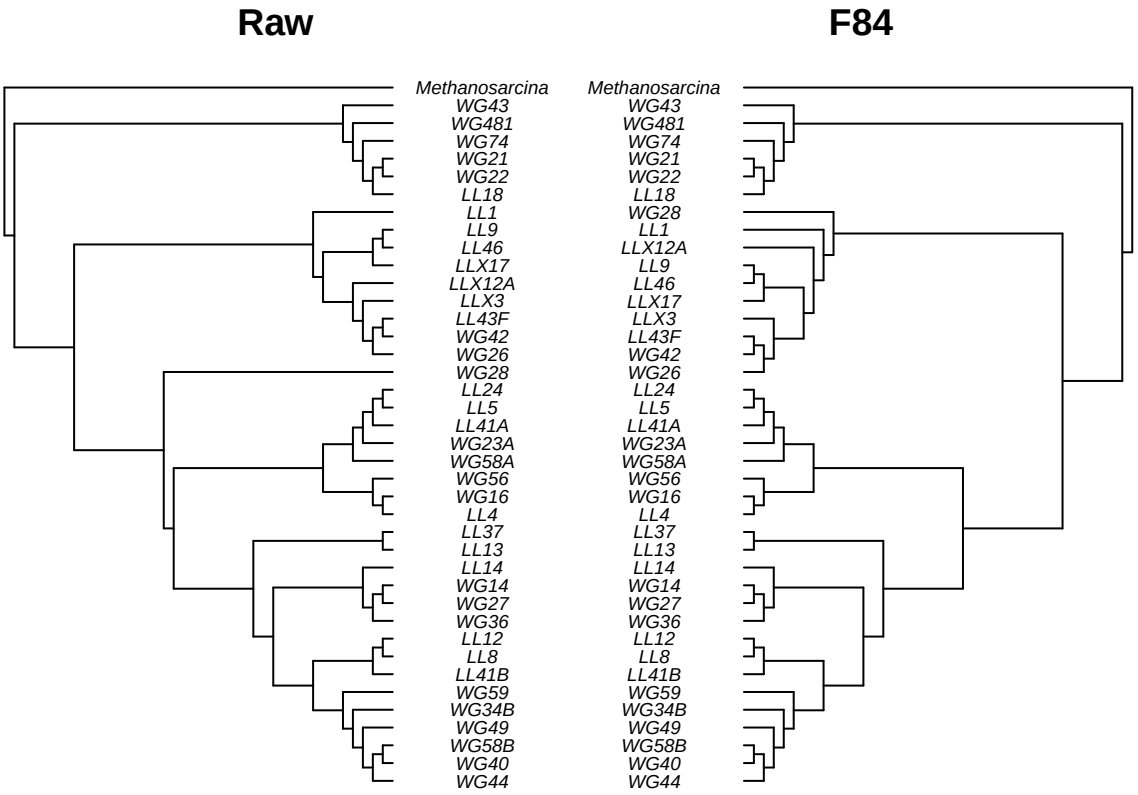
Now, let's compare the "raw" and "F84" distance matrices. First, we will make a **saturation plot**, which allow us to visuazlie the effect of our DNA subsititution model compared to a distance matrix that does not include this feature.

```
# Plot Distances from Different DNA Substitution Models
par(mar = c(5, 5, 2, 1) + 0.1)
plot(seq.dist.raw, seq.dist.F84,
     pch = 20, col = "red", las = 1, asp = 1, xlim = c(0, 0.7), ylim = c(0, 0.7),
     xlab = "Raw Distance", ylab = "F84 Distance")
abline(b = 1, a = 0, lty = 2)
text(0.65, 0.6, "1:1")
```



Second, we will assess how the "raw" and "F84" distance matrices affect tree topology by making a **cophylogenetic plot**. This will give us a visual sense of whether or not the two trees are in agreement with one another.

```
# Make Neighbor Joining Trees Using Different DNA Substitution Models {ape}
raw.tree <- bionj(seq.dist.raw)
F84.tree <- bionj(seq.dist.F84)

# Define Outgroups
raw.outgroup <- match("Methanosarcina", raw.tree$tip.label)
F84.outgroup <- match("Methanosarcina", F84.tree$tip.label)

# Root the Trees {ape}
raw.rooted <- root(raw.tree, raw.outgroup, resolve.root=TRUE)
F84.rooted <- root(F84.tree, F84.outgroup, resolve.root=TRUE)

# Make Cophylogenetic Plot {ape}
layout(matrix(c(1,2), 1, 2), width = c(1, 1))
par(mar = c(1, 1, 2, 0))
plot.phylo(raw.rooted, type = "phylogram", direction = "right", show.tip.label=TRUE,
           use.edge.length = FALSE, adj = 0.5, cex = 0.6, label.offset = 2, main = "Raw")

par(mar = c(1, 0, 2, 1))
plot.phylo(F84.rooted, type = "phylogram", direction = "left", show.tip.label=TRUE,
           use.edge.length = FALSE, adj = 0.5, cex = 0.6, label.offset = 2, main = "F84")
```



Question 3: Using the saturation plot and cophylogenetic plots from above, describe the effect that the F84 substitution model has on our phylogenetic reconstruction. If the plots seem to be inconsistent with one another, explain what it giving rise to the differences.

Answer 3:

5) INTEGRATING TRAITS AND PHYLOGENY

In the context of biodiveristy, many researchers are interested in being able to map traits onto phylogenetic trees. This can be important for addressing questions related to diversification, trait evolution, and ecological interactions. In this section, we will import functional trait data that was collected on the 39 freshwater isoaltes for which we have 16S rRNA sequences. Specifically, we measured the growth rates of each strain on 18 different forms of phosphorus (e.g., phosphate, DNA, ATP, etc.). With this information we will visualize the functional traits (i.e., growth on different phosphorus) in a phylogenetic conetxt by mapping them onto the neighbor joining tree (with the F84 DNA substituion model) that we created.

A. Loading Trait Database

```
# Import Growth Rate Data
p.growth <- read.table("./data/p.isolates.raw.growth.txt", sep = "\t", header = TRUE,
                       row.names = 1)

# Standadize Growth Rates Across Strains
p.growth.std <- p.growth / (apply(p.growth, 1, sum))
```

B. Trait Manipulations

From the section above where we described the data, you will recall that we were interested in testing for a **generalist-specialist trade-off**. To do this, we need to create two new variables. First, we need calculate the maximum growth rate (μ_{max}) of each isoalte across all phosphorus types. This will help us test the expectation that generalists will have lower maximum growth rates because there is a cost associated with being able to use a lot of different chemical forms of phosphorus. In contrast, we expect that specialists will have a high maximum growth rate on their preferred phosphorus resource.

```
# Calculate Max Growth Rate
umax <- (apply(p.growth, 1, max))
```

Second, we need to come up with a way of quantifying whether a strain is a generalist or a specialist. We will use the niche breadth (nb) index from Levins (1968) which is defined as $\frac{1}{(n \cdot \Sigma p_{xi}^2)}$, where n is the total number of resources and p_{xi} is the proportion of observed growth for isolate i on each resource (x).

```
levins <- function(p_xi = ""){
  p = 0
  for (i in p_xi){
    p = p + i^2
    }
  nb = 1 / (length(p_xi) * p)
  return(nb)
  }
```

Let's apply our `nb()` function to the isolate growth rate data on the different phosphorus resources.

```
# Calculate Niche Breadth for Each Isolate
nb <- as.matrix(levins(p.growth.std))

# Add Row & Column Names to Niche Breadth Matrix
rownames(nb) <- row.names(p.growth)
colnames(nb) <- c("NB")
```

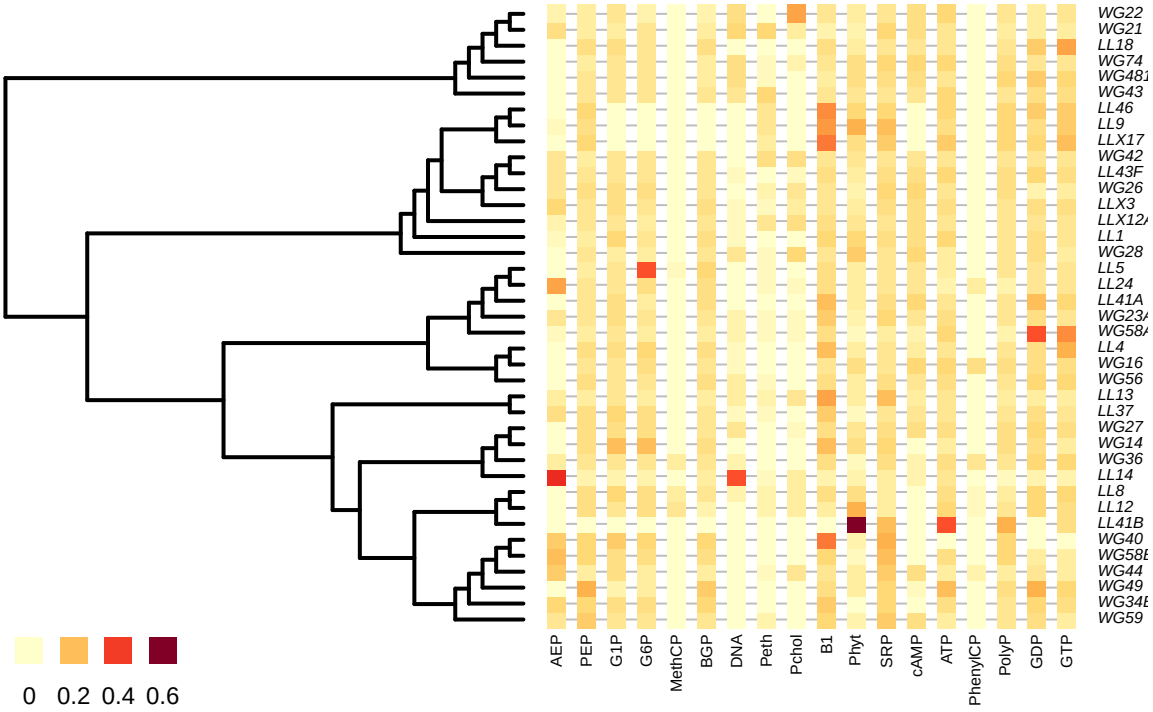
C. Visualzing Traits on Trees

Now that we have identified and calculated our traits of interest, we can map those traits onto our phylogenetic tree in order to observe any major patterns. This is similar to the hypothesis-generating visualizations we did in other lessons this semsester. Before we start, there are a few things that need to be done. First, we need to be sure that we are dealing with the correct version of the tree, so we will recreate the tree. Second, we need to make sure that this tree is rooted. Last, we need to remove the root, because this isn't part of our traits analysis.

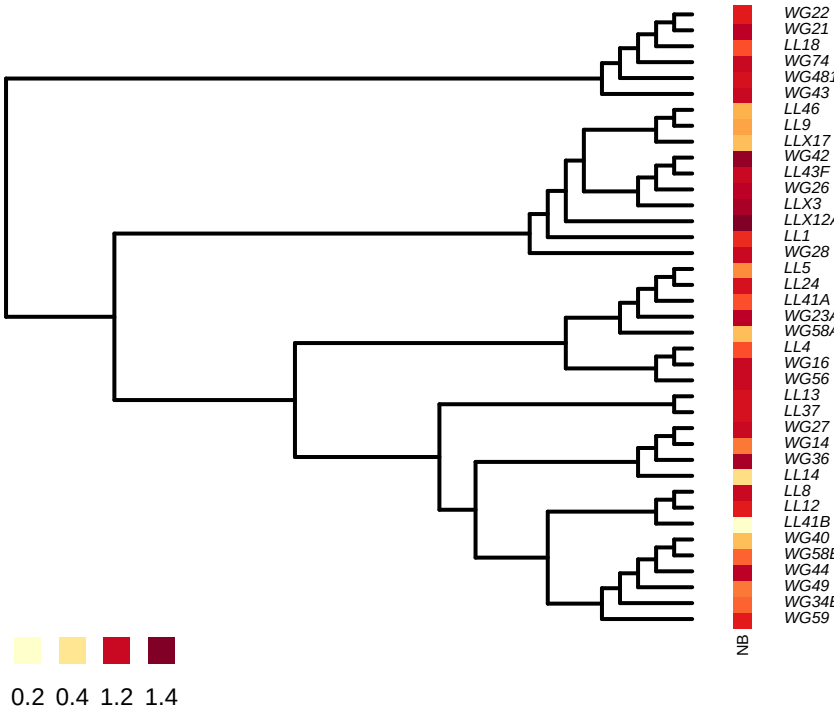
```
# Generate Neighbor Joining Tree Using F84 DNA Substitution Model {ape}
nj.tree <- bionj(seq.dist.F84)

# Define the Outgroup
outgroup <- match("Methanosarcina", nj.tree$tip.label)

# Create a Rooted Tree {ape}
nj.rooted <- root(nj.tree, outgroup, resolve.root = TRUE)

# Keep Rooted but Drop Outgroup Branch
nj.rooted <- drop.tip(nj.rooted, "Methanosarcina")
```

Now we can plot our traits onto the tree. This can be done with either a group of traits, such as the growth rates on different phosphorus sources, or it can be done on a single trait, such as niche breadth. Here we will do both.

```
# Define Color Palette
mypalette <- colorRampPalette(brewer.pal(9, "YlOrRd"))

# Map Phosphorus Traits {adephylo}
par(mar=c(1,1,1,1) + 0.1)
x <- phylo4d(nj.rooted, p.growth.std)
table.phylo4d(x, treetype = "phylo", symbol = "colors", show.node = TRUE,
              cex.label = 0.5, scale = FALSE, use.edge.length = FALSE,
              edge.color = "black", edge.width = 2, box = FALSE,
              col=mypalette(25), pch = 15, cex.symbol = 1.25,
              ratio.tree = 0.5, cex.legend = 1.5, center = FALSE)
```




```
# Niche Breadth
par(mar=c(1,5,1,5) + 0.1)
x.nb <- phylo4d(nj.rooted, nb)
table.phylo4d(x.nb, treetype = "phylo", symbol = "colors", show.node = TRUE,
              cex.label = 0.5, scale = FALSE, use.edge.length = FALSE,
              edge.color = "black", edge.width = 2, box = FALSE,
              col=mypalette(25), pch = 15, cex.symbol = 1.25, var.label=("       NB"),
              ratio.tree = 0.90, cex.legend = 1.5, center = FALSE)
```



Question 4: Based on the distribution of traits on the neighbor joining tree that we created, what might you predict about the degree of specialization of bacteria on diverse phosphorus resources?

Answer 4:

6) HYPOTHESIS TESTING

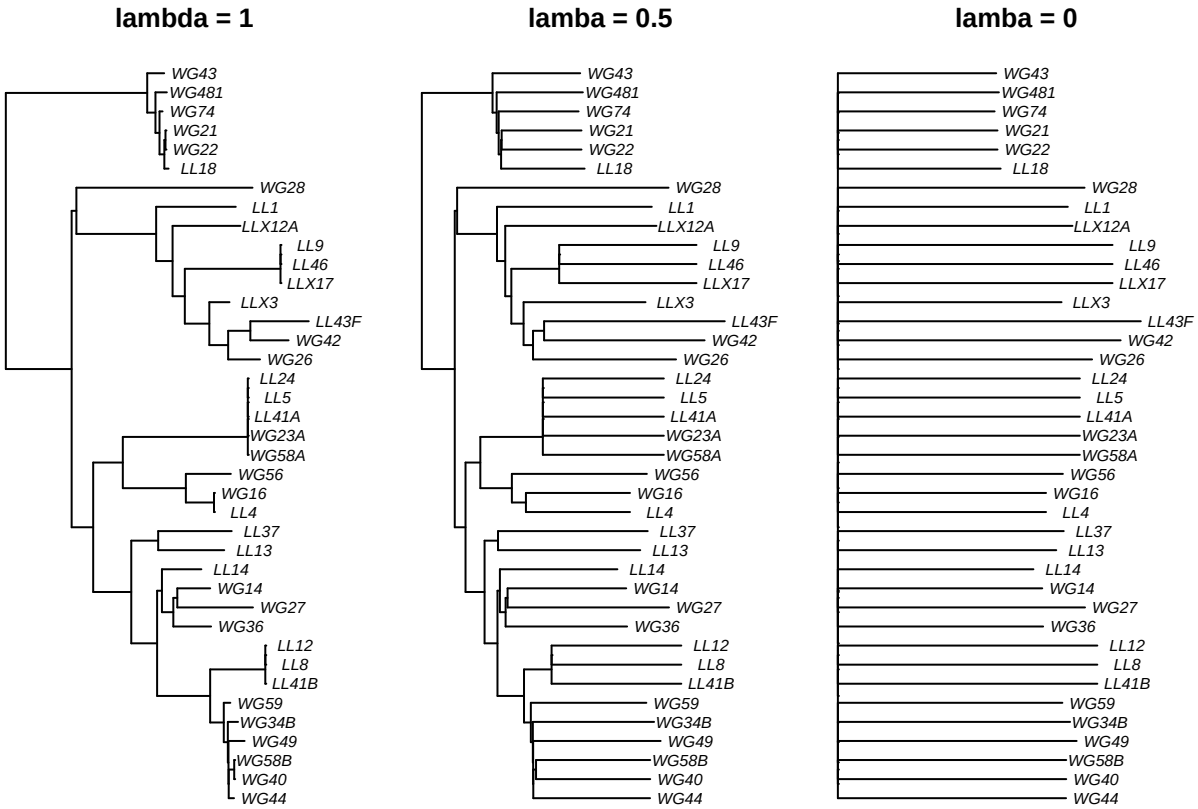
An emerging goal of biodiversity research is to understand the effect traits have on the peformance of species under different environmental conditions, but also how traits can affect ecosytem functioning. Before attempting to establish relationships among traits, however, it is important to understand the role evolution plays in the distribution of traits. Owing to shared ancestry, we expect species traits to be non-independent with respect to evolutionary history. Therefore, it is important that we test for the presence of this **phylogenetic signal** before drawing conclusions about patterns among traits. In addition, these patterns of phylogenetic signal provide insight into ecolgoical and evoutionary processes giving rise to the **clustering** or **overdispersion** of traits, which are sometimes associated with enviornmental filtering and competitive repulsion, respectivley. In the following sections, we introduce some methods for testing hypotheses about the distribution of traits with respect to phylogeny.

A) Phylogenetic Signal: Pagel's Lambda

Mark Pagel developed a fairly simple technique to quantify phylogenetic signal, which involves a tree transformation (http://goo.gl/zSSxoB). The transformation is accomplished by scaling the off-diagonal elements of the variance-covariance matrix, which describe the tree topology and branch lengths, by the value "lambda". When lambda equals 1, you have your original, non-transformed branch lengths. When lambda equals 0, you have removed all phylogentic signal from your tree. Let's look at what happens when we transform our tree with lambda values of 1, 0.5, and 0.

```
# Visualize Trees With Different Levels of  Phylogenetic Signal {geiger}
nj.lambda.5 <- rescale(nj.rooted, "lambda", 0.5)
nj.lambda.0 <- rescale(nj.rooted, "lambda", 0)
```

```
layout(matrix(c(1,2,3), 1, 3), width = c(1, 1, 1))
par(mar=c(1,0.5,2,0.5)+0.1)
plot(nj.rooted, main = "lambda = 1", cex = 0.7, adj = 0.5)
plot(nj.lambda.5, main = "lamba = 0.5", cex = 0.7, adj = 0.5)
plot(nj.lambda.0, main = "lamba = 0",  cex = 0.7, adj = 0.5)
```



Now, let's generate quantitative output that will allow us to assess how much phylogenetic signal is in our data set. For this test, we will compare the phylogenetic signal of niche breadth using trees with lambda values of 1 (untransformed) and 0 (transformed).

```
# Generate Test Statistics for Comparing Phylogenetic Signal {geiger}
fitContinuous(nj.rooted, nb, model = "lambda")
fitContinuous(nj.lambda.0, nb, model = "lambda")
```

Question 5: There are two important outputs from the `fitContinuous` function that can help you interpret the phylogenetic signal in trait data sets. First, compare the lambda values of the untransformed tree to the transformed (lambda = 0). Second, compare the Akaike information criterion (AIC) scores of the two models. Differences in AIC scores provide maximum likelihood inference regarding the quality of model fit to a given data set. Models with lower AIC values are better. However, if the difference in AIC values between two models (Δ AIC) isn't greater than at least 2, then then the models are considered equivalent. With this information in hand, what does Pagel's lambda say about the phylogenetic signal of niche breadth in our data set?

Answer 5:

B) Phylogenetic Signal: Blomberg's K

Blomberg et al. (2003) derived a statistic that quantifies phylogenetic signal by comparing observed trait distributions on a tree to what would evolve under Browninan (i.e., random) motion (http://goo.gl/0xU9Et).

Blomberg's K is calculated as the mean squared error of the tip data (i.e., trait) measured from the phylogenetic-corrected mean and the mean squared error based on the variance–covariance matrix derived from the given phylogeny under the assumption of Browninan motion (Münkemüller et al 2012, http://goo.gl/c08XrA). A K-value of 1 means that a trait is distributed on the tree according to null expectation of Browning motion. A K-value > 1 means that traits are **clustered** with closely related species being *more similar* than expected by chance. A K-value of < 1 means that traits are **overdispersed** with closely related species *less similar* than expected by chance.

Let's test for phylogenetic signal using Blomberg's K for the standardized growth of our bacterial isolates on *each phosphorus source* based on our neighbor joining tree.

Note: In this particular analysis, we are generating 18 test statistics with associated p-values (one for each phosphorus resource). This means that we are increasing our probability of rejecting the null hypothesis (ie. "no phylogenetic signal") when it is in fact true. This is called **false discovery rate** (FDR). We are going to correct for FDR using the Benjamini-Hochberg method ("PIC.P.BH").

```
# First, Correct for Zero Branch-Lengths on Our Tree
nj.rooted$edge.length <- nj.rooted$edge.length + 10^-7

# Calculate Phylogenetic Signal for Growth on All Phosphorus Resources
# First, Create a Blank Output Matrix
p.phylosignal <- matrix(NA, 6, 18)
colnames(p.phylosignal) <- colnames(p.growth.std)
rownames(p.phylosignal) <- c("K", "PIC.var.obs", "PIC.var.mean",
                             "PIC.var.P", "PIC.var.z", "PIC.P.BH")

# Use a For Loop to Calculate Blomberg's K for Each Resource
for (i in 1:18){
  x <- as.matrix(p.growth.std[ ,i, drop = FALSE])
  out <- phylosignal(x, nj.rooted)
  p.phylosignal[1:5, i] <- round(t(out), 3)
}

# Use the BH Correction on P-values:
p.phylosignal[6, ] <- round(p.adjust(p.phylosignal[4, ], method = "BH"), 3)
```

In addition, let's test for phylogenetic signal using Blomberg's K on *niche breadth*:

```
# Calcualate Phylogenetic Signal for Niche Breadth
signal.nb <- phylosignal(nb, nj.rooted)
```

Question 6: Using the K-values and associated p-values (i.e., "PIC.var.P"") from the `phylosignal` output, answer the following questions:

a. Is there significant phylogenetic signal for niche breadth or standardized growth on any of the phosphorus resources
b. If there is significant phylogenetic signal, are the results suggestive of clustering or overdispersion?

Answer 6a:
Answer 6b:

C. Calculate Dispersion of a Trait

Another way to calculate the phylogenetic signal of a trait is to use a formal test of dispersion. In ecology, dispersion is commonly used to meaasure of how things (e.g., traits or species) are distrubuted in space. However, we can use similar approaches to determine how well traits are distributed across a phylogenetic tree. Fritz & Purvis (2010) derived a measure of dispersion (D) for categorical traits (http://goo.gl/SUrm5j). In

our case study, we can use D to deteremine if the ability of a bacterial isolate to grow on a specific phosphorus resrouce is overdispersed or clustered. Here, we will calcuate D using the `phylo.d()` function in the `caper` package.

First, there are a few things we need to do: 1) turn our continuous growth data into categorical data, 2) add a column to our data set for our isolate names, and 3) combine our tree and trait data using the `comparative.data()` command in `caper`.

The output of phylo.d() will return the estimated value of D. Estimates of D can be negative (clustered) or positive (overdispersed). When D is close 0, traits are randomly clumped. When D is close to 1, traits are dispersed in way that is consistent with Brownian motion.

The `phylo.d()` function uses a permutaion test to calculate the probability of D being different from 1 (no phylogenetic structure; random) or 0 (Brownian phylogenetic structure). Let's use this function to calculate dispersion (D) on a few of our phosphorus traits.

```
# Turn Continuous Data into Categorical Data
p.growth.pa <- as.data.frame((p.growth > 0.01) * 1)

# Look at Phosphorus Use for Each Resource
apply(p.growth.pa, 2, sum)

# Add Names Column to Data
p.growth.pa$name <- rownames(p.growth.pa)

# Merge Trait and Phylogenetic Data; Run `phylo.d`
p.traits <- comparative.data(nj.rooted, p.growth.pa, "name")
phylo.d(p.traits, binvar = AEP)
phylo.d(p.traits, binvar = PhenylCP)
phylo.d(p.traits, binvar = DNA)
phylo.d(p.traits, binvar = cAMP)
```

Question 7: Using the estimates for D and the probabilities of each phylogenetic model, answer the following questions:

a. Choose three phosphorus growth traits and test whether they are significantly clustered or overdispersed?

b. How do these results compare the results from the Blomberg's K analysis?

c. Discuss what factors might give rise to differences between the metrics.

Answer 7a:
Answer 7b:
Answer 7c:

7) QUANTIFYING TRAIT EVOLUTION

talk about how things are related, need to correct for

A. Correcting for the underlying phylogeny

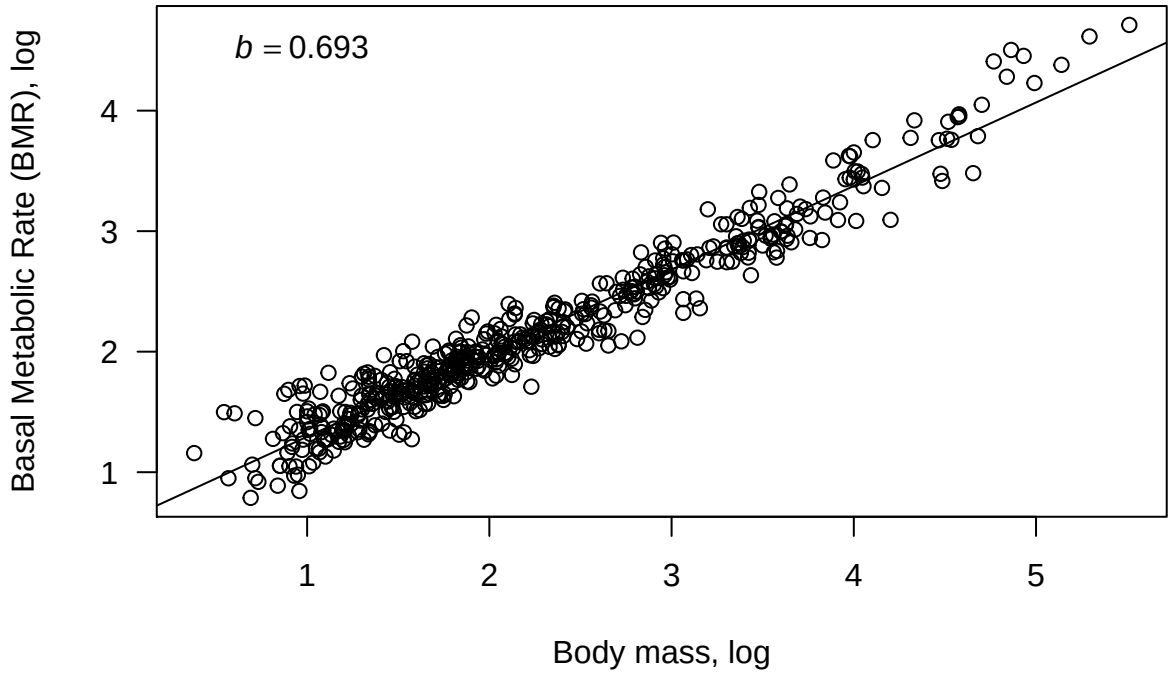
Clean mammal dataset

```
mammal.Tree <- read.tree("./data/mammal_best_super_tree_fritz2009.tre")
mammal.data <- read.table("./data/mammal_BMR.txt", sep = "\t", header = TRUE)
# subset
```

```
mammal.data<- mammal.data[,c("Species","BMR_.mlO2.hour.","Body_mass_for_BMR_.gr.")]
mammal.species <- array(mammal.data$Species)
pruned.mammal.tree <- drop.tip(mammal.Tree, mammal.Tree$tip.label[-na.omit(match(mammal.species, mammal

pruned.mammal.data <- mammal.data[mammal.data$Species %in% pruned.mammal.tree$tip.label,]
# turn column of Species names into rownames
rownames(pruned.mammal.data) <- pruned.mammal.data$Species
```

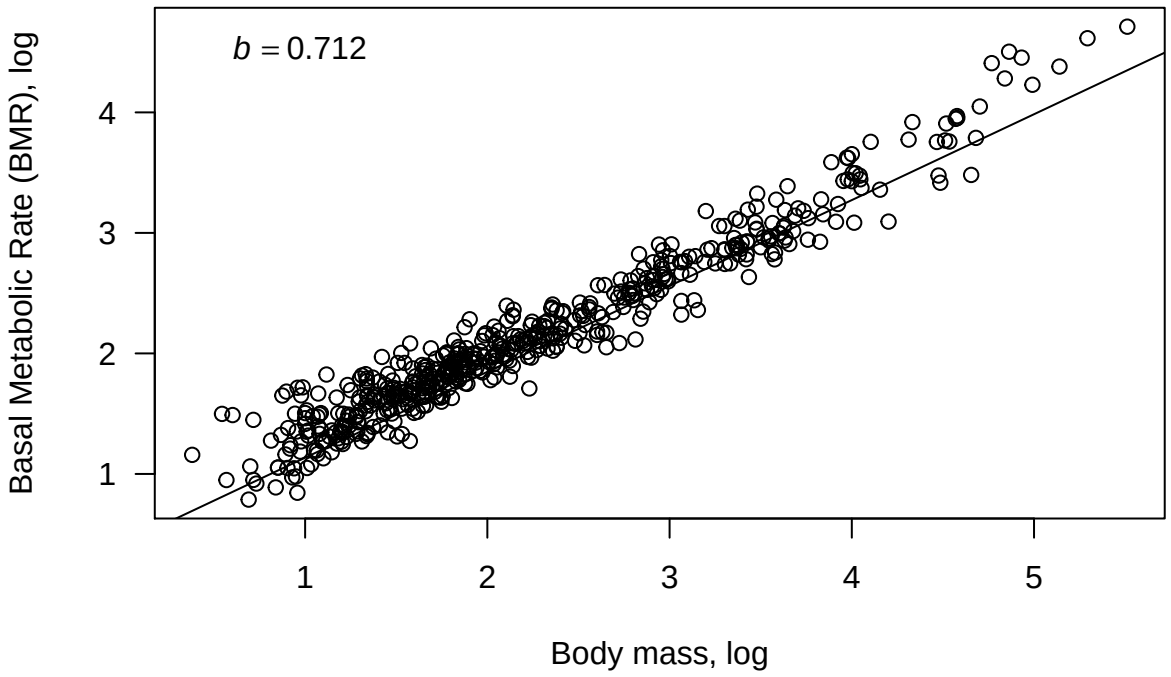
Let's look at the relationship between mass and BMR

```
fit <- lm(log10(BMR_.mlO2.hour.) ~ log10(Body_mass_for_BMR_.gr.), data=pruned.mammal.data)
plot(log10(pruned.mammal.data$Body_mass_for_BMR_.gr.), log10(pruned.mammal.data$BMR_.mlO2.hour.),  las =
abline(a = fit$coefficients[1], b = fit$coefficients[2])
b1 <- round(fit$coefficients[2],3)
eqn <- bquote(italic(b) == .(b1))
text(0.5, 4.5, eqn, pos = 4)
```



Now let's correct for phylogeny

```
fit.phy <- phylolm(log10(BMR_.mlO2.hour.) ~ log10(Body_mass_for_BMR_.gr.), data = pruned.mammal.data,
                   pruned.mammal.tree, model = 'lambda')

plot(log10(pruned.mammal.data$Body_mass_for_BMR_.gr.), log10(pruned.mammal.data$BMR_.mlO2.hour.),  las =
abline(a = fit.phy$coefficients[1], b = fit.phy$coefficients[2])
b1.phy <- round(fit.phy$coefficients[2],3)
eqn <- bquote(italic(b) == .(b1.phy))
text(0.5, 4.5, eqn, pos = 4)
```



B.

REFERENCES

Boettiger, C., Coop, G., & Ralph, P. (2012). Is your phylogeny informative? Measuring the power of comparative methods. Evolution, 66(7), 2240–2251. http://doi.org/10.1111/j.1558-5646.2011.01574.x

Bartoszek, K., Pienaar, J., Mostad, P., Andersson, S., & Hansen, T. F. (2012). A phylogenetic comparative method for studying multivariate adaptation. Journal of Theoretical Biology, 314, 204–215. http://doi.org/10.1016/j.jtbi.2012.08.005

Butler, M., & King, A. A. (2004). Phylogenetic comparative analysis: a modeling approach for adaptive evolution. American Naturalist, 164(6), 683–695. Revell, L. J. (2010). Phylogenetic signal and linear regression on species data. Methods in Ecology and Evolution, 1(4), 319–329. http://doi.org/10.1111/j.2041-210X.2010.00044.x

Cadotte, M. W., & Davies, T. J. (2016). Phylogenies in Ecology: A Guide to Concepts and Methods. Princeton, NJ: Princeton University Press.

Lynch, M. (1991). Methods for the Analysis of Comparative Data in Evolutionary Biology. Evolution, 45(5), 1065. http://doi.org/10.2307/2409716

Paradis, E. (2012). Analysis of Phylogenetics and Evolution with R. New York: Springer. Hansen, T. F., & Bartoszek, K. (2012). Interpreting the evolutionary regression: The interplay between observational and biological errors in phylogenetic comparative studies. Systematic Biology, 61(3), 413–425. http://doi.org/10.1093/sysbio/syr122