8. Worksheet: Phylogenetic Diversity - Traits

Anna Werkowski; Z620: Quantitative Biodiversity, Indiana University

22 February, 2023

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 is often necessary to 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 be able to:

1. create phylogenetic trees to view evolutionary relationships from sequence data
2. map functional traits onto phylogenetic trees to visualize the distribution of traits with respect to evolutionary history
3. test for phylogenetic signal within trait distributions and trait-based patterns of biodiversity

Directions:

1. In the Markdown version of this document in your cloned repo, change "Student Name" on line 3 (above) with your name.
2. Complete as much of the worksheet as possible during class.
3. Use the handout as a guide; it contains a more complete description of data sets along with examples of proper scripting needed to carry out the exercises.
4. Answer questions in the worksheet. Space for your answers is provided in this document and is indicated by the ">" character. If you need a second paragraph be sure to start the first line with ">". You should notice that the answer is highlighted in green by RStudio (color may vary if you changed the editor theme).
5. Before you leave the classroom today, it is *imperative* that you **push** this file to your GitHub repo, at whatever stage you are. This will enable you to pull your work onto your own computer.
6. When you have completed the worksheet, **Knit** the text and code into a single PDF file by pressing the `Knit` button in the RStudio scripting panel. This will save the PDF output in your '8.BetaDiversity' folder.
7. After Knitting, please submit the worksheet by making a **push** to your GitHub repo and then create a **pull request** via GitHub. Your pull request should include this file (**11.PhyloTraits_Worksheet.Rmd**) with all code blocks filled out and questions answered) and the PDF output of `Knitr` (**11.PhyloTraits_Worksheet.pd**

The completed exercise is due on **Wednesday, February 22$^{\text{nd}}$, 2023 before 12:00 PM (noon)**.

1) SETUP

Typically, the first thing you will do in either an R script or an RMarkdown file is setup your environment. This includes things such as setting the working directory and loading any packages that you will need.

In the R code chunk below, provide the code to:
1. clear your R environment,
2. print your current working directory,
3. set your working directory to your "*/8.PhyloTraits*" folder, and
4. load all of the required R packages (be sure to install if needed).

```
rm(list = ls())
getwd()
```

```
## [1] "/Users/annawerkowski/Documents/Github/QB2023_Werkowski/2.Worksheets/8.PhyloTraits"
```

```
setwd("/Users/annawerkowski/Documents/Github/QB2023_Werkowski/2.Worksheets/8.PhyloTraits")

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)
  }
}
```

```
##
## Attaching package: 'seqinr'

## The following objects are masked from 'package:ape':
##
##     as.alignment, consensus

##
## Attaching package: 'phylobase'

## The following object is masked from 'package:ape':
##
##     edges

##
## Attaching package: 'permute'

## The following object is masked from 'package:seqinr':
##
##     getType

## This is vegan 2.6-4

##
## Attaching package: 'nlme'

## The following object is masked from 'package:seqinr':
##
##     gls
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select

## The following object is masked from 'package:nlme':
##
##     collapse

## The following object is masked from 'package:seqinr':
##
##     count

## The following object is masked from 'package:ape':
##
##     where

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

##
## Attaching package: 'phangorn'

## The following objects are masked from 'package:vegan':
##
##     diversity, treedist

##
## Attaching package: 'phylotools'

## The following object is masked from 'package:seqinr':
##
##     read.fasta

## Registered S3 method overwritten by 'dendextend':
##   method     from
##   rev.hclust vegan

##
## ---------------------
## Welcome to dendextend version 1.16.0
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'
```

```
## The following object is masked from 'package:permute':
##
##     shuffle
```

```
## The following object is masked from 'package:geiger':
##
##     is.phylo
```

```
## The following objects are masked from 'package:phylobase':
##
##     labels<-, prune
```

```
## The following objects are masked from 'package:ape':
##
##     ladderize, rotate
```

```
## The following object is masked from 'package:stats':
##
##     cutree
```

```
##
## Attaching package: 'phylogram'
```

```
## The following object is masked from 'package:dendextend':
##
##     prune
```

```
## The following object is masked from 'package:phylobase':
##
##     prune
```

```
##
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:geiger':
##
##     rescale
```

```r
if (!require("BiocManager", quietly = TRUE)){
  install.packages("BiocManager")
}
BiocManager::install("msa")
```

```
## Bioconductor version 3.16 (BiocManager 1.30.19), R 4.2.1 (2022-06-23)
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   `force = TRUE` to re-install: 'msa'
```

```
## Old packages: 'RcppArmadillo', 'readstata13'
```

```r
BiocManager::install('muscle')
```

```
## Bioconductor version 3.16 (BiocManager 1.30.19), R 4.2.1 (2022-06-23)
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   `force = TRUE` to re-install: 'muscle'
```

```
## Old packages: 'RcppArmadillo', 'readstata13'
```

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 perform exceptionally well on a particular resource compared to the performance of a generalist. 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

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

> ***Answer 1***: The p.isolates.afa file contained lots of dashes and all uppercase letters. The p.isolates.fasta file contained no dashes, some spaces, and all lowercase letters. The fasta file was visually better and seemed much less chaotic.

In the R code chunk below, do the following: 1. read your alignment file, 2. convert the alignment to a DNAbin object, 3. select a region of the gene to visualize (try various regions), and 4. plot the alignment using a grid to visualize rows of sequences.

```r
#import and view unaligned sequences
#seqs <- readDNAStringSet("/Users/annawerkowski/Documents/Github/QB2023_Werkowski/2.Worksheets/8.PhyloT
#align sequences using MUSCLE parameters
#read.aln <- msaMuscle(seqs)
#save and export the alignment to use later
#save.aln <- msaConvert(read.aln, type = "bios2mds::align")
#export.fasta(save.aln, "./data/p.isolates.afa")
```

```
#convert alignment to DNAbin object
#p.DNAbin <- as.DNAbin(read.aln)
#identify base pair region of 16s rRNA gene to visualize
#window <- p.DNAbin[, 100:500]
#visualize sequence alignment
#image.DNAbin(window, cex.lab = 0.50)
```

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

a. Approximately how long are our sequence reads?

b. What regions do you think would are appropriate for phylogenetic inference and why?

> ***Answer 2a***: They are approximately 400 bp long.
> ***Answer 2b***: The regions where there are all of the same color lined up would be good for continuing with phylogenetic inference. This woould be good because this shows that they are aligned and that we can now evaluate the shared evolutionary history between them.

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.

A. Neighbor Joining Trees

In the R code chunk below, do the following:
1. calculate the distance matrix using `model = "raw"`,
2. create a Neighbor Joining tree based on these distances,
3. define "Methanosarcina" as the outgroup and root the tree, and
4. plot the rooted tree.

```
#create a distance matrix with "raw" model
#seq.dist.raw <- dist.dna(p.DNAbin, model = "raw", pairwise.deletion = FALSE)

#neighbor joining algorithm to construct tree, a phylo object
#nj.tree <- bionj(seq.dist.raw)
#identify outgroup sequence
#outgroup <- match("Methanosarcina", nj.tree$tip.label)
#root the tree
#nj.rooted <- root(nj.tree, outgroup, resolve.root = TRUE)
#plot the rooted tree
#par(mar = c(1,1,2,1) + 0.1)
#plot.phylo(nj.rooted, main = "Neighbor Joining Tree", "phylogram", use.edge.length = FALSE, direction
#add.scale.bar(cex = 0.7)
```

Question 3: What are the advantages and disadvantages of making a neighbor joining tree?

> ***Answer 3***: Advantage – it create a guide tree that can be used as a base that more sophisticated models can be added onto. Disadvantage – It may not account for multiple substitutions that may have occured within the base pairs in the same site over time.

B) SUBSTITUTION MODELS OF DNA EVOLUTION

In the R code chunk below, do the following:
1. make a second distance matrix based on the Felsenstein 84 substitution model,
2. create a saturation plot to compare the *raw* and *Felsenstein (F84)* substitution models,
3. make Neighbor Joining trees for both, and
4. create a cophylogenetic plot to compare the topologies of the trees.

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

#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
#abline(b = 1, a = 0, lty = 2)
#text(0.65, 0.6, "1:1")
```

In the R code chunk below, do the following:
1. pick another substitution model,
2. create a distance matrix and tree for this model,
3. make a saturation plot that compares that model to the *Felsenstein (F84)* model,
4. make a cophylogenetic plot that compares the topologies of both models, and
5. be sure to format, add appropriate labels, and customize each plot.

```
#make neighbor joining trees using different DNA substitution models
#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
#raw.rooted <- root(raw.tree, raw.outgroup, resolve.root = TRUE)
#F84.rooted <- root(F84.tree, F84.outgroup, resolve.root = TRUE)
#make cophylogenetic plot
#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
#par(mar = c(1,0,2,1))
#plot.phylo(F84.rooted, type = "phylogram", direction = "left", show.tip.label = TRUE, use.edge.length
```

```
#seq.dist.T92 <- dist.dna(p.DNAbin, model = "T92", pairwise.deletion = FALSE)
#make neighbor joining trees using different DNA substitution models
#F84.tree <- bionj(seq.dist.F84)
#T92.tree <- bionj(seq.dist.T92)
#define outgroups
#F84.outgroup <- match("Methanosarcina", F84.tree$tip.label)
#T92.outgroup <- match("Methanosarcina", T92.tree$tip.label)
#root the trees
#T92.rooted <- root(T92.tree, T92.outgroup, resolve.root = TRUE)
#F84.rooted <- root(F84.tree, F84.outgroup, resolve.root = TRUE)
#make cophylogenetic plot
#layout(matrix(c(1,2), 1, 2), width = c(1,1))
#par(mar = c(1,1,2,0))
```

```
#plot.phylo(T92.rooted, type = "phylogram", direction = "right", show.tip.label = TRUE, use.edge.length
#par(mar = c(1,0,2,1))
#plot.phylo(F84.rooted, type = "phylogram", direction = "left", show.tip.label = TRUE, use.edge.length
```

```
#quantifying phylogenetic similarity
#set method = "PH85" for the symmetric difference
#set method = "score" for the symmetric difference
#this function automatically checks for a root and unroots rooted trees
#can then pass it either the rooted or unrooted tree and get same answer
#dist.topo(raw.rooted, F84.rooted, method = "score")
```

Question 4:

a. Describe the substitution model that you chose. What assumptions does it make and how does it compare to the F84 model?
b. Using the saturation plot and cophylogenetic plots from above, describe how your choice of substitution model affects your phylogenetic reconstruction. If the plots are inconsistent with one another, explain why.
c. How does your model compare to the *F84* model and what does this tell you about the substitution rates of nucleotide transitions?

> ***Answer 4a***: I chose the T92 model (Tamura Model). It assumes equal frequencies of nucleotides but recognizes that transition mutations occur with higher probability than transversion mutations. It also accounts for G + C content. ***Answer 4b***: The substitution model that you choose will change the look of your phylogenetic reconstruction depending upon what assumptions the model has. The T92 and F84 models seem to be quite robust. ***Answer 4c***: I was surprised to find that the F84 and T92 models were identical.I believe that this tells me that the substitution rates of nucleotide transitions are the same between these two models.

C) ANALYZING A MAXIMUM LIKELIHOOD TREE

In the R code chunk below, do the following:
1. Read in the maximum likelihood phylogenetic tree used in the handout. 2. Plot bootstrap support values onto the tree

```
#requires alignment to be read in with as phyDat object
#phyDat.aln <- msaConvert(read.aln, type = "phangorn::phyDat")
#make the NJ tree for the maximum likelihood method
#Phangorn requires a specific attribute class, so we remake the trees
#aln.dist <- dist.ml(phyDat.aln)
#aln.NJ <- NJ(aln.dist)
#fit <- pml(tree = aln.NJ, data = phyDat.aln)
#fit tree using the JC69 substitution model
#fitJC <- optim.pml(fit, TRUE)
#fit tree using a GTR model with gamma distributed rates
#fitGTR <- optim.pml(fit, model = "GTR", optInv = TRUE, optGamma = TRUE, rearrangement = "NNI", control
#perform model selection with either an ANOVA or AIC
#anova(fitJC, fitGTR)
#AIC(fitJC)
#AIC(fitGTR)
```

```
#bootstrap support
#ml.bootstrap <- read.tree("./data/ml_tree/RAxML_bipartitions.T1")
#par(mar = c(1,1,2,1) + 0.1)
#plot.phylo(ml.bootstrap, type = "phylogram", direction = "right", show.tip.label = TRUE, use.edge.leng
#add.scale.bar(cex = 0.7)
#nodelabels(ml.bootstrap$node.label, font = 2, bg = "white", frame = "r", cex = 0.5)
```

Question 5:

a) How does the maximum likelihood tree compare the to the neighbor-joining tree in the handout? If the plots seem to be inconsistent with one another, explain what gives rise to the differences.

b) Why do we bootstrap our tree?

c) What do the bootstrap values tell you?

d) Which branches have very low support?

e) Should we trust these branches?

> ***Answer 5a***: The maximum likelihood tree is very similar to the neighbor joining tree but is a more robust and statistically correct evaluation of the phylogeny. Neighbor joining does not take into account specific nucleotide states while the maximum likelihood does. The two graphs are inconsistent with each other and that could be because of the amount of error that can be included in a neighbor joining one versus the maximum likelihood. In this case, it would be better to accept the maximum likelihood.
> ***Answer 5b***: We bootstrap the tree to reassess how reliable our tree is. By bootstrapping, we can determine whether the nodes of the tree are operationally correct. ***Answer 5c***: Higher bootstrap values indicate that they are operationally correct while lower values provide insufficient support.
> ***Answer 5d***: The branches at the top of the tree (from a vertical standpoint), WG42 to LL1, are the lowest. The further down you go vertically, the more operationally correct the values are.
> ***Answer 5e***: No we should not.

5) INTEGRATING TRAITS AND PHYLOGENY

A. Loading Trait Database

In the R code chunk below, do the following:
1. import the raw phosphorus growth data, and
2. standardize the data for each strain by the sum of growth rates.

```
#import trait database
#p.growth <- read.table("./data/p.isolates.raw.growth.txt", sep = "\t", header = TRUE, row.names = 1)
#standardize growth rates across strains
#p.growth.std <- p.growth / (apply(p.growth, 1, sum))
```

B. Trait Manipulations

In the R code chunk below, do the following:
1. calculate the maximum growth rate (μ_{max}) of each isolate across all phosphorus types,
2. create a function that calculates niche breadth (nb), and
3. use this function to calculate nb for each isolate.

```
#calculate max growth rate
#umax <- (apply(p.growth, 1, max))
#quantify whether strains are specialist or generalist
#levins <- function(p_xi = ""){
  #p = 0
  #for (i in p_xi){
    #p = p + i^2
  #}
  #nb = 1 / (length(p_xi) * p)
  #return(nb)
#}

#calculate niche breadth for each isolate
#nb <- as.matrix(levins(p.growth.std))
#add row names to niche breadth matrix
#nb <- setNames(as.vector(nb), as.matrix(row.names(p.growth)))
```

C. Visualizing Traits on Trees

In the R code chunk below, do the following:
1. pick your favorite substitution model and make a Neighbor Joining tree,
2. define your outgroup and root the tree, and
3. remove the outgroup branch.

```
#generate neighbor joining tree using F84 DNA substitution model
#nj.tree <- bionj(seq.dist.F84)
#define the outgroup
#outgroup <- match("Methanosarcina", nj.tree$tip.label)
#create a rooted tree
#nj.rooted <- root(nj.tree, outgroup, resolve.root = TRUE)
#keep rooted but drop outgroup branch
#nj.rooted <- drop.tip(nj.rooted, "Methanosarcina")
#plot to look at new tree
#plot(nj.rooted)
```

In the R code chunk below, do the following:
1. define a color palette (use something other than “YlOrRd”),
2. map the phosphorus traits onto your phylogeny,
3. map the *nb* trait on to your phylogeny, and
4. customize the plots as desired (use `help(table.phylo4d)` to learn about the options).

```
#define color palette
#mypalette <- colorRampPalette(brewer.pal(9, "YlOrRd"))
#correct for zero branch lengths on the tree
#nj.plot <- nj.rooted
#nj.plot$edge.length <- nj.plot$edge.length + 10^-1
#map phosphorus traits
#par(mar = c(1,1,1,1) + 0.1)
#x <- phylo4d(nj.plot, p.growth.std)
#table.phylo4d(x, treetype = "phylo", symbol = "colors", show.node = TRUE, cex.label = 0.5, scale = FAL

#niche breadth
```

```
#par(mar = c(1,5,1,5) + 0.1)
#x.nb <- phylo4d(nj.plot, nb)
#table.phylo4d(x.nb, treetype = "phylo", symbol = "colors", show.node = TRUE, cex.label = 0.5, scale = 
```

Question 6:

a) Make a hypothesis that would support a generalist-specialist trade-off.

b) What kind of patterns would you expect to see from growth rate and niche breadth values that would support this hypothesis?

> ***Answer 6a***: Generalist species have a wide preference for many things and therefore have a large niche breadth. Specialist species have a specific preference which limits the breadth of their niche. There is a trade off between the preferences that they have and the niche breadth that they can occupy. ***Answer 6b***: In their preferred conditions, generalists will grow to their umax. They have many resources available to them due to their niche breadth. In the same way, specialists will also grow to their own umax in their preferred conditions. This is assuming niche partitioning which allows species to partition their resources, utilize their own preferences, and maintain equilibrium.

6) HYPOTHESIS TESTING

A) Phylogenetic Signal: Pagel's Lambda

In the R code chunk below, do the following:
1. create two rescaled phylogenetic trees using lambda values of 0.5 and 0,
2. plot your original tree and the two scaled trees, and
3. label and customize the trees as desired.

```
#visualize trees with different levels of phylogenetic signal
#nj.lambda.5 <- geiger::rescale(nj.rooted, "lambda", 0.5)
#nj.lambda.0 <- geiger::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.rooted, main = "lambda = 0.5", cex = 0.7, adj = 0.5)
#plot(nj.rooted, main = "lambda = 0", cex = 0.7, adj = 0.5)
```

In the R code chunk below, do the following:
1. use the `fitContinuous()` function to compare your original tree to the transformed trees.

```
#generate test statistics for comparing phylogenetic signal
#fitContinuous(nj.rooted, nb, model = "lambda")
```

```
#fitContinuous(nj.lambda.0, nb, model = "lambda")
```

```
#compare pagel's lambda score with likelihood ratio test
#when lambda = 0, no phylogenetic signal
#phylosig(nj.rooted, nb, method = "lambda", test = TRUE)
```

Question 7: There are two important outputs from the `fitContinuous()` function that can help you interpret the phylogenetic signal in trait data sets. a. Compare the lambda values of the untransformed tree to the transformed (lambda = 0). b. Compare the Akaike information criterion (AIC) scores of the two models. Which model would you choose based off of AIC score (remember the criteria that the difference in AIC values has to be at least 2)? c. Does this result suggest that there's phylogenetic signal?

> ***Answer 7a***: The lambda value of the untransformed tree was 0.006975 while the transformed tree had a value of 0. ***Answer 7b***: To be honest, the AIC values for both models were incredibly close. I would choose the transformed model AIC values only because they were slightly lower. ***Answer 7c***: I ran the phylosig for Pagel's lambda and the value was not 0. If it had been 0, we would say that there is no phylogenetic signal. However, there is one there.

B) Phylogenetic Signal: Blomberg's K

In the R code chunk below, do the following:
1. correct tree branch-lengths to fix any zeros,
2. calculate Blomberg's K for each phosphorus resource using the `phylosignal()` function,
3. use the Benjamini-Hochberg method to correct for false discovery rate, and
4. calculate Blomberg's K for niche breadth using the `phylosignal()` function.

```
#correct for zero branch-lengths on our tree
#nj.rooted$edge.length <- nj.rooted$edge.length + 10^-7
#calculate pylogenetic signal for growth on all phosphorus resources
#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")
#create a for loop to calculate Blomberg's K for each resource
#for (i in 1:18){
  #x <- setNames(as.vector(p.growth.std[,i]), row.names(p.growth))
  #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)
#check out the results
#print(p.phylosignal)
```

```
# calculate phylogenetic signal for niche breadth
#signal.nb <- phylosignal(nb, nj.rooted)
#print(signal.nb)
```

Question 8: 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 8a***: There is no significant phylogenetic signal for niche breadth or standardized growth. ***Answer 8b***: There was no significant signal found.

C. Calculate Dispersion of a Trait

In the R code chunk below, do the following:
1. turn the continuous growth data into categorical data,
2. add a column to the data with the isolate name,
3. combine the tree and trait data using the `comparative.data()` function in `caper`, and
4. use `phylo.d()` to calculate D on at least three phosphorus traits.

```
#turn continuous data into categorical data
#p.growth.pa <- as.data.frame((p.growth > 0.01) * 1)
#look at phosphorus 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, permut = 10000)
#phylo.d(p.traits, binvar = PhenylCP, permut = 10000)
#phylo.d(p.traits, binvar = DNA, permut = 10000)
#phylo.d(p.traits, binvar = cAMP, permut = 10000)
```

Question 9: 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 9a***: Three growth traits are AEP, PhenylCP, and DNA. All of them were positive values, indicating overdispersion. For AEP, the values were closer to 0, so the traits were randomly clustered. For PhenylCP, the values for random were closer to 1, indicating Brownian motion. The values for Brownian were closer to 0, indicating random clumping. For DNA, all of the values were close to 0, indicating random clumping. ***Answer 9b***: The Blomberg's K value was incredibly small while these values are mildly small. ***Answer 9c***: I have honestly no idea as to what would cause this difference. It could potentially be because of differences in robustness of tests.

7) PHYLOGENETIC REGRESSION

In the R code chunk below, do the following:
1. Load and clean the mammal phylogeny and trait dataset, 2. Fit a linear model to the trait dataset, examining the relationship between mass and BMR, 2. Fit a phylogenetic regression to the trait dataset, taking into account the mammal supertree

```
#correspondence between evolutionary history and ecology
#generate Jaccard's index (dissimiliarity)
#noNo <- vegdist(p.growth.pa[,1:18], method = 'jaccard', binary = TRUE)

#test the clustering method that best fits that data
```

```
#define linkage methods
#m <- c("average", "single", "complete", "ward")
#names(m) <- c("average", "single", "complete", "ward")

#function to compute agglomerative coefficient
#ac <- function(x){
  #agnes(noNo, method = x)$ac
#}

#calculate agglomerative coefficient for each clustering linkage method
#use the method with the highest coefficient, indicates better fit
#sapply(m, ac)

#generate heirarchical cluster
#no.tree <- hclust(noNo, method = "ward.D2")
#plot(no.tree)
```

##PHYLOGENETIC REGRESSION##

```
#nb.lake <- as.data.frame(as.matrix(nb))
#nb.lake$lake = rep('A')

#for(i in 1:nrow(nb.lake)) {
  #ifelse(grepl("LL",row.names(nb.lake)[i]), nb.lake[i,2] <- "LL",nb.lake[i,2]<-"WG")
#}

#add a meaningful column name to the niche breadth values
#colnames(nb.lake)[1] <- "NB"
#calculate the max growth rate
#umax <- as.matrix((apply(p.growth,1, max)))
#nb.lake = cbind(nb.lake, umax)

#plot maximum growth rate by niche breadth
#ggplot(data = nb.lake, aes(x = NB, y = log10(umax), color = lake)) +
  #geom_point() +
  #geom_smooth(method = "lm") +
  #xlab("Niche Breadth") +
  #ylab(expression(Log[10]~"(Maximum Growth Rate)"))
```

```
#simple linear regression
#fit.lm <- lm(log10(umax) ~ NB*lake, data = nb.lake)
#summary(fit.lm)
```

```
#AIC(fit.lm)
```

Phylogeny-Corrected Regression

```
#fit.plm <- phylolm(log10(umax) ~ NB*lake, data = nb.lake, nj.rooted, model = "lambda", boot = 0)
#summary(fit.plm)
```

##QUESTION 10## a. Why do we need to correct for shared evolutionary history? b. How does a phylogenetic regression differ from a standard linear regression? c. Interpret the slope and fit of each model. Did accounting for shared evolutionary history improve or worsen the fit? d. Try to come up with a scenario where the relationship between two variables would completely disappear when the underlying phylogeny is accounted for.

> ***Answer 10a***: We need to correct for shared evolutionary history because if we were to not correct then there would be too many variables that could cause error/changes. By correcting for that, you know that the differences you see are because of differences in other variables. ***Answer 10b***: In a standard linear regression, residual errors are assumed to be independent and identically distributed variables that are normally distributed. In a phylogenetic regression the residual errors are described by a covariance matrix that takes into account the phylogeny that established the branches.So in the phylogenetic regression, the variables are not assumed to be independent. ***Answer 10c***: The fit of the phylogenetic model was much better than the general linear regression. ***Answer 10d***: The first thing I think of when I think of this is the difference between homology and synapomorphy. We assume that traits from different species are related because of homology but when the underlying phylogeny is accounted for we can determine that these are not related at all. We teach this in L111 when talking about dolphins and killer whales.

7) SYNTHESIS

Work with members of your Team Project to obtain reference sequences for taxa in your study. Sequences for plants, animals, and microbes can found in a number of public repositories, but perhaps the most commonly visited site is the National Center for Biotechnology Information (NCBI) https://www.ncbi.nlm.nih.gov/. In almost all cases, researchers must deposit their sequences in places like NCBI before a paper is published. Those sequences are checked by NCBI employees for aspects of quality and given an **accession number**. For example, here an accession number for a fungal isolate that our lab has worked with: JQ797657. You can use the NCBI program nucleotide **BLAST** to find out more about information associated with the isolate, in addition to getting its DNA sequence: https://blast.ncbi.nlm.nih.gov/. Alternatively, you can use the `read.GenBank()` function in the `ape` package to connect to NCBI and directly get the sequence. This is pretty cool. Give it a try.

But before your team proceeds, you need to give some thought to which gene you want to focus on. For microorganisms like the bacteria we worked with above, many people use the ribosomal gene (i.e., 16S rRNA). This has many desirable features, including it is relatively long, highly conserved, and identifies taxa with reasonable resolution. In eukaryotes, ribosomal genes (i.e., 18S) are good for distinguishing course taxonomic resolution (i.e. class level), but it is not so good at resolving genera or species. Therefore, you may need to find another gene to work with, which might include protein-coding gene like cytochrome oxidase (COI) which is on mitochondria and is commonly used in molecular systematics. In plants, the ribulose-bisphosphate carboxylase gene (*rbcL*), which on the chloroplast, is commonly used. Also, non-protein-encoding sequences like those found in **Internal Transcribed Spacer (ITS)** regions between the small and large subunits of of the ribosomal RNA are good for molecular phylogenies. With your team members, do some research and identify a good candidate gene.

After you identify an appropriate gene, download sequences and create a properly formatted fasta file. Next, align the sequences and confirm that you have a good alignment. Choose a substitution model and make a tree of your choice. Based on the decisions above and the output, does your tree jibe with what is known about the evolutionary history of your organisms? If not, why? Is there anything you could do differently that would improve your tree, especially with regard to future analyses done by your team?

```
#group project work is included in a separate file
```