8. Worksheet: Phylogenetic Diversity - Traits

Student Name; Z620: Quantitative Biodiversity, Indiana University

26 February, 2025

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, **push** this file to your GitHub repo.
6. For the assignment portion of the worksheet, follow the directions at the bottom of this file.
7. When you are done, **Knit** the text and code into a PDF file.
8. After Knitting, submit the completed exercise by creating a **pull request** via GitHub. Your pull request should include this file `PhyloTraits_Worskheet.Rmd` and the PDF output of `Knitr` (`PhyloTraits_Worskheet.pdf`).

The completed exercise is due on **Wednesday, February 26th, 2025 before 12:00 PM (noon)**.

1) SETUP

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 `Week6-PhyloTraits/` folder, and
4. load all of the required R packages (be sure to install if needed).

```
getwd()
```

```
## [1] "C:/Users/ADMIN/OneDrive/Documents/GitHub/QB2025_Nambiar/Week6-PhyloTraits"
```

```
setwd("C:/Users/ADMIN/OneDrive/Documents/GitHub/QB2025_Nambiar/Week6-PhyloTraits/data")
```

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 main difference is that p.isolates.afa has dashes inserted (show as -) so that all sequences are the same length - aligned at matching positions(I think), whereas p.isolates.fasta has the orgigal sequences without those gaps.

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.

```
package.list <- c("ape", "seqinr", "phylobase", "adephylo", "geiger",
                  "picante", "stats", "RColorBrewer", "caper", "phylolm", "pmc",
                  "ggplot2", "tidyr", "dplyr", "phangorn", "pander", "phytools",
                  "vegan", "cluster", "dendextend", "phylogram", "bios2mds",
                  "pak", "formatR")

for (package in package.list) {
  if (!require(package, character.only = TRUE, quietly = TRUE)) {
    install.packages(package)
    library(package, character.only = TRUE)
  }
}
```

```
## Warning: package 'ape' was built under R version 4.4.2
```

```
## Warning: package 'seqinr' was built under R version 4.4.2
```

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

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

```
## Warning: package 'phylobase' was built under R version 4.4.2

##
## Attaching package: 'phylobase'

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

## Warning: package 'adephylo' was built under R version 4.4.2

## Warning: package 'ade4' was built under R version 4.4.2

## Warning: package 'geiger' was built under R version 4.4.2

## Warning: package 'phytools' was built under R version 4.4.2

## Warning: package 'maps' was built under R version 4.4.1

##
## Attaching package: 'phytools'

## The following object is masked from 'package:phylobase':
##
##     readNexus

## Warning: package 'picante' was built under R version 4.4.2

## Warning: package 'vegan' was built under R version 4.4.2

## Warning: package 'permute' was built under R version 4.4.2

##
## Attaching package: 'permute'

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

## This is vegan 2.6-8

##
## Attaching package: 'vegan'

## The following object is masked from 'package:phytools':
##
##     scores

##
## Attaching package: 'nlme'
```

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

## Warning: package 'caper' was built under R version 4.4.2

## Warning: package 'mvtnorm' was built under R version 4.4.1

## Warning: package 'phylolm' was built under R version 4.4.2

## Warning: package 'pmc' was built under R version 4.4.2

##
## 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

## Warning: package 'phangorn' was built under R version 4.4.2

##
## Attaching package: 'phangorn'

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

## Warning: package 'pander' was built under R version 4.4.2
```

```
##
## Attaching package: 'cluster'

## The following object is masked from 'package:maps':
##
##     votes.repub

## Warning: package 'dendextend' was built under R version 4.4.2

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

##
## --------------------
## Welcome to dendextend version 1.19.0
## 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 object is masked from 'package:phytools':
##
##     untangle

## 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

## Warning: package 'phylogram' was built under R version 4.4.2

##
## Attaching package: 'phylogram'

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

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

## Warning: package 'bios2mds' was built under R version 4.4.2

## Warning: package 'amap' was built under R version 4.4.1

##
## Attaching package: 'amap'

## The following object is masked from 'package:vegan':
##
##     pca

## Warning: package 'e1071' was built under R version 4.4.1

##
## Attaching package: 'scales'

## The following object is masked from 'package:phytools':
##
##     rescale

## Warning: package 'rgl' was built under R version 4.4.1

## Warning: package 'pak' was built under R version 4.4.2

## Warning: package 'formatR' was built under R version 4.4.2
```

```
# Comment out after first run to prevent continuous installation
# Reinstall/update pak: pkg_install("msa")

library(msa)
```

```
## Warning: package 'msa' was built under R version 4.4.1
```

```
## Loading required package: Biostrings

## Warning: package 'Biostrings' was built under R version 4.4.2

## Loading required package: BiocGenerics

## Warning: package 'BiocGenerics' was built under R version 4.4.1

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union

## The following object is masked from 'package:ade4':
##
##     score

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
##     get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, saveRDS, setdiff,
##     table, tapply, union, unique, unsplit, which.max, which.min

## Loading required package: S4Vectors

## Warning: package 'S4Vectors' was built under R version 4.4.1

## Loading required package: stats4

##
## Attaching package: 'S4Vectors'

## The following objects are masked from 'package:dplyr':
##
##     first, rename

## The following object is masked from 'package:tidyr':
##
##     expand
```

```
## The following object is masked from 'package:utils':
##
##     findMatches

## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname

## Loading required package: IRanges

## Warning: package 'IRanges' was built under R version 4.4.1

##
## Attaching package: 'IRanges'

## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice

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

## The following object is masked from 'package:grDevices':
##
##     windows

## Loading required package: XVector

## Warning: package 'XVector' was built under R version 4.4.1

## Found more than one class "Annotated" in cache; using the first, from namespace 'RNeXML'

## Also defined by 'S4Vectors'

## Found more than one class "Annotated" in cache; using the first, from namespace 'RNeXML'

## Also defined by 'S4Vectors'

## Found more than one class "Annotated" in cache; using the first, from namespace 'RNeXML'

## Also defined by 'S4Vectors'

## Found more than one class "Annotated" in cache; using the first, from namespace 'RNeXML'

## Also defined by 'S4Vectors'

## Found more than one class "Annotated" in cache; using the first, from namespace 'RNeXML'
```

```
## Also defined by 'S4Vectors'

## Found more than one class "Annotated" in cache; using the first, from namespace 'RNeXML'

## Also defined by 'S4Vectors'

## Loading required package: GenomeInfoDb

## Warning: package 'GenomeInfoDb' was built under R version 4.4.1

##
## Attaching package: 'Biostrings'

## The following object is masked from 'package:dendextend':
##
##     nnodes

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

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

## The following object is masked from 'package:base':
##
##     strsplit
```

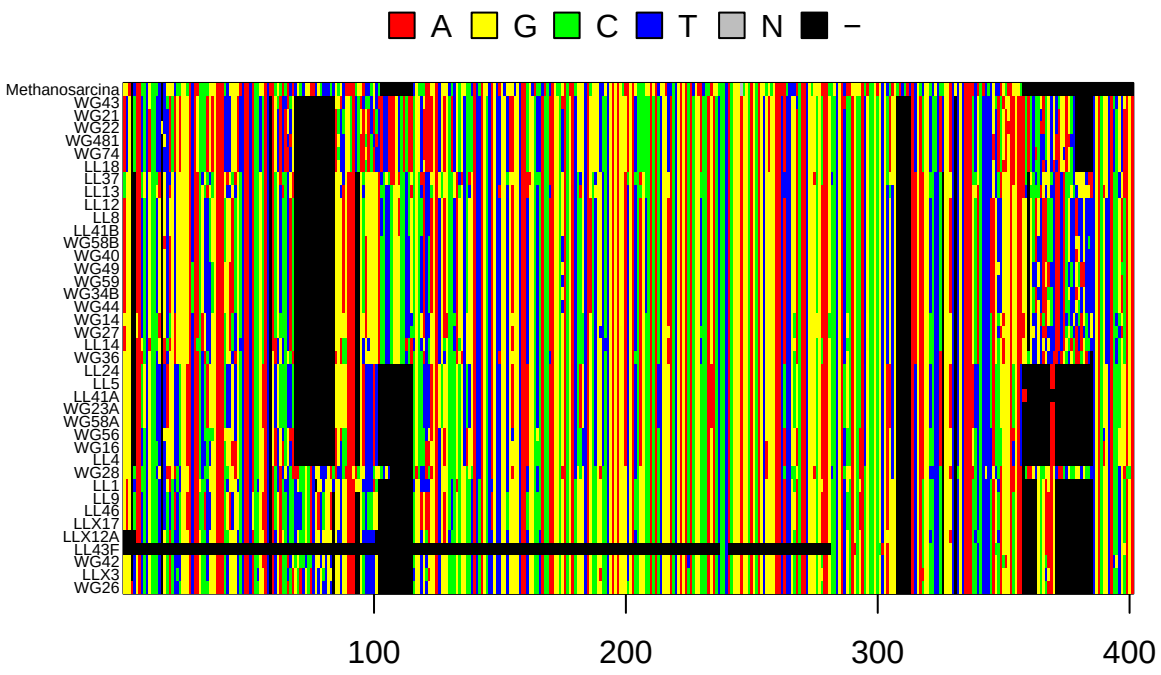
```
#reading the FASTA files

seq1<-readDNAStringSet("data/p.isolates.fasta", format='fasta')
seq1
```

```
## DNAStringSet object of length 40:
##      width seq                                               names
##  [1]   619 ACACGTGAGCAATCTGCCCTTCT...TTCTCTGGGAATACCTGACGCT LL9
##  [2]   597 CGGCAGCGGGAAGTAGCTTGCTA...AACTGTTCAGCTAGAGTCTTGT WG14
##  [3]   794 CAGCGGCGGACGGGTGAGTAACA...GCTAACGCATTAAGCACTCCGC WG28
##  [4]   716 CTTCAGAGTTAGTGGCGGACGGG...TGCTAGTTGTCGGGATGCATGC LL24
##  [5]   803 ACGAACTCTTCGGAGTTAGTGGC...TAAAACTCAAAGGAATTGACGG LL41A
##  ...   ... ...
## [36]   652 TTCGGGAGTACACGAGCGGCGAA...TTCTCTGGGAATACCTGACGCT LL46
## [37]   661 GCGAACGGGTGAGTAACACGTGG...GAGCGAAAGCGTGGGTAGCGAA WG26
## [38]   694 GGCGAACGGGTGAGTAACACGTG...ACCCTGGTAGTCCACGCCGTAA WG42
## [39]   699 TACAGGTACCAGGCTCCTTCGGG...AAAGCATGGGTAGCGAACAGGA LLX17
## [40]  1426 TTCTGGTTGATCCTGCCAGAGGT...AACCTNAATTTTGCAAGGGGGG Methanosarcina
```

```
seq2<-readDNAStringSet("data/p.isolates.afa", format='fasta')
seq2
```

```
## DNAStringSet object of length 40:
##      width seq                                                names
##  [1]  1500 TTCTGGTTGATCCTGCCAGAGGT...AACCTNAATTTTGCAAGGGGGG Methanosarcina
##  [2]  1500 ------------------------...---------------------- WG43
##  [3]  1500 ------------------------...---------------------- WG21
##  [4]  1500 ------------------------...---------------------- WG22
##  [5]  1500 ------------------------...---------------------- WG481
##  ...   ... ...
## [36]  1500 ------------------------...---------------------- LLX12A
## [37]  1500 ------------------------...---------------------- LL43F
## [38]  1500 ------------------------...---------------------- WG42
## [39]  1500 ------------------------...---------------------- LLX3
## [40]  1500 ------------------------...---------------------- WG26
```

```
read.aln<-msaMuscle(seq1)
#convert alignment to DNAbin object {ape}
p.DNAbin<-as.DNAbin(read.aln)

#identify base pair region of 16S rRNA gene to visualize
window<-p.DNAbin[, 100:500]

#command to visualize the sequence alignemnt
image.DNAbin(window, cex.lab=0.5)
```



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***: When we run seq1 we see that bacterial isolates range from ~600–800 bp. The outgroup Methanosarcina is ~1400 bp long. ***Answer 2b***: The 16S rRNA gene regions are the best choicesicne there are highly conserved yet still variable enough to differentiate betweenspecies.

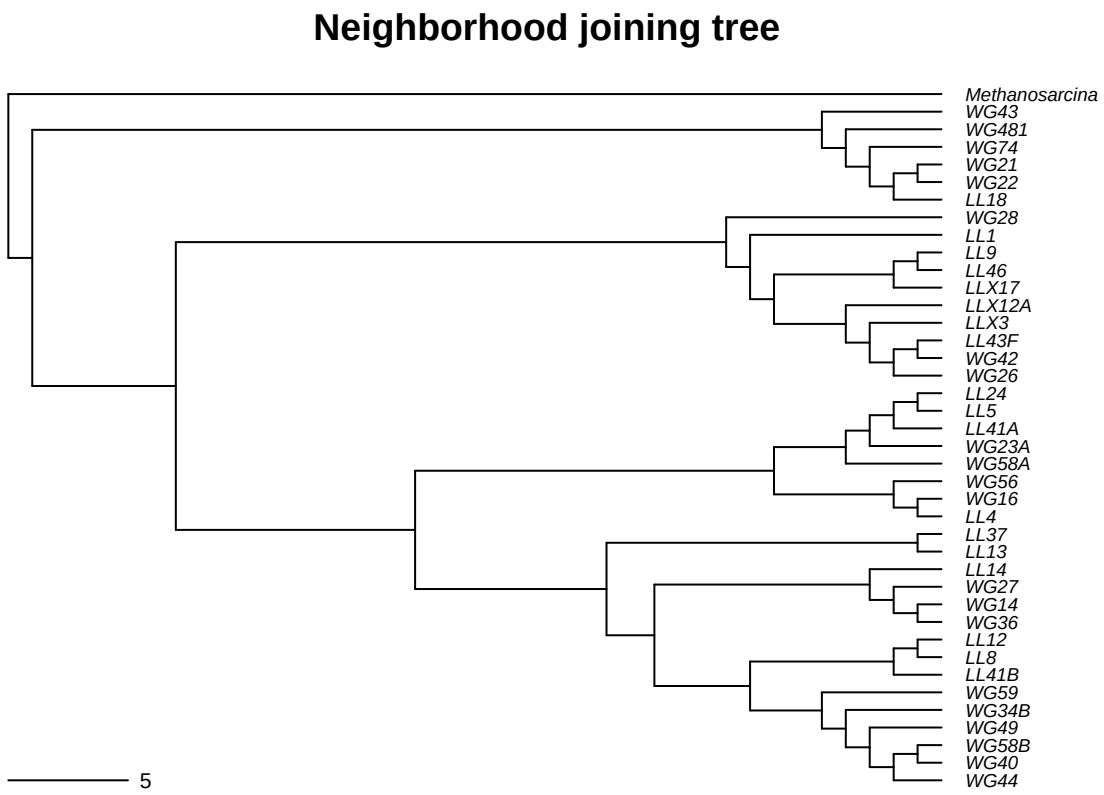
4) MAKING A PHYLOGENETIC TREE

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 distance matrix with "raw" model {ape}

seq.dist.raw<-dist.dna(p.DNAbin, model="raw", pairwise.deletion=FALSE)

#neighbor joining algorithm to construct tree,

nj.tree<-bionj(seq.dist.raw)

# identify otugroup sequence
outgroup<-match("Methanosarcina", nj.tree$tip.label)

#root the tree

nj.rooted<-root(nj.tree, outgroup, resolve.root=TRUE)

#plot  the rooted
par(mar=c(1,1,2,1)+ 0.1)
plot.phylo(nj.rooted,main="Neighborhood joining tree", "phylogram",
           use.edge.length=FALSE, direction="right", cex =0.6 ,
           label.offset = 1)

add.scale.bar(cex=0.7)
```



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

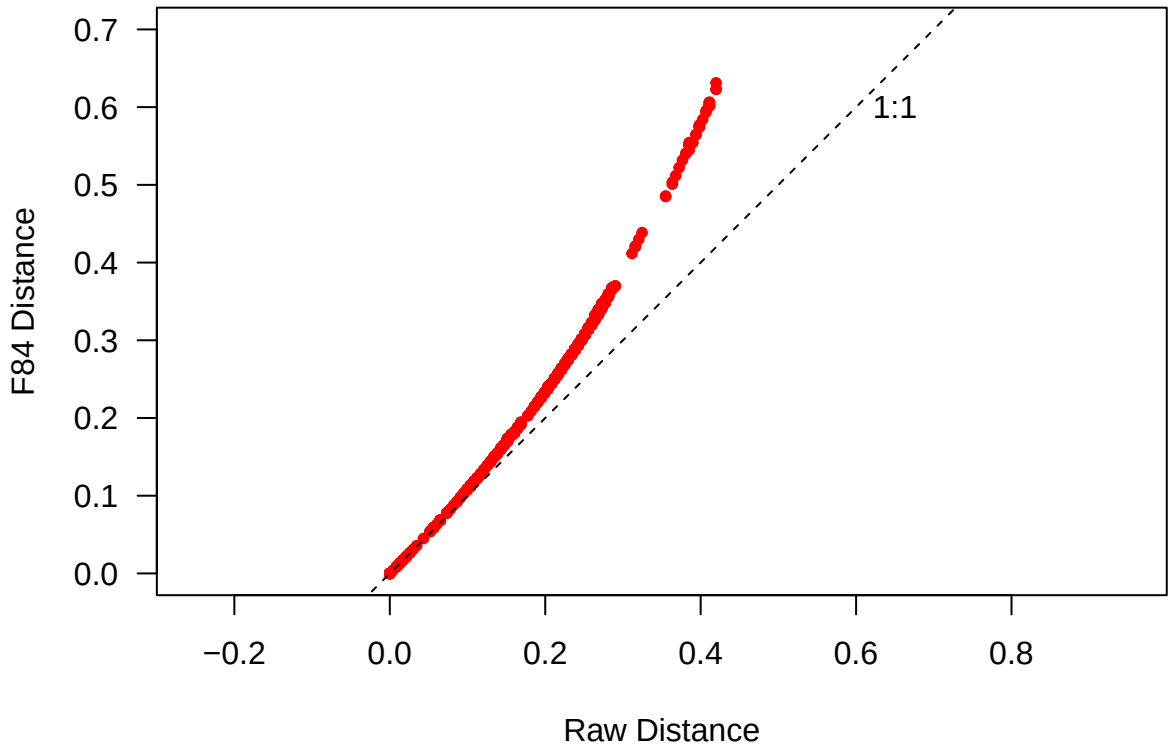
> ***Answer 3***: The neighborhood joining approach is considered to be a simple and reliable way to make a tree. It is also used as a "guide tree". As I understand, the issue with this meothd would be that relying ona distance matrix will not contain information on how the DNA evolved over time. Since it only produces one tree it cannot be used as a statistical method.

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.

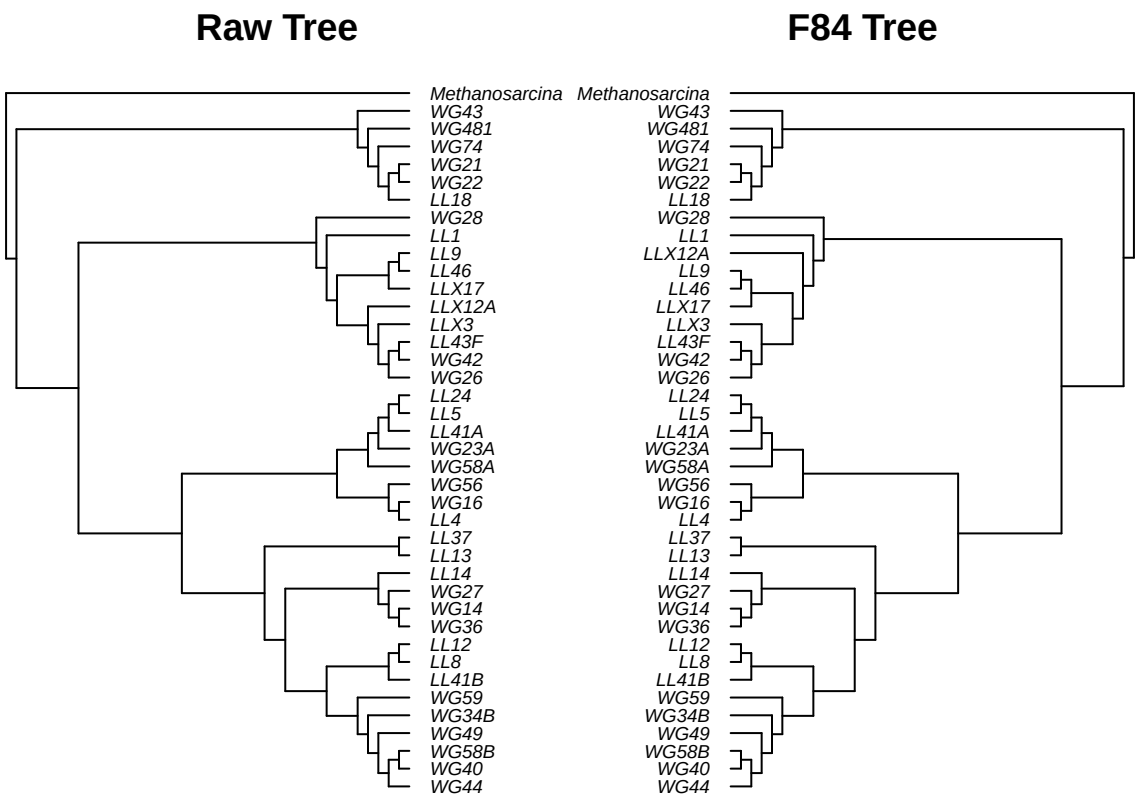
```
#create distance matrix with the "F84" model
seq.dist.F84<-dist.dna(p.DNAbin, model="F84", pairwise.deletion= FALSE)

# 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, 0.7),
     xlab = "Raw Distance", ylab = "F84 Distance")
```

```
abline(b = 1, a = 0, lty = 2)

text(0.65, 0.6, "1:1")
```




```
# 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 = FALSE,
           cex = 0.6, label.offset = 2, main = "Raw Tree")
```

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




```
#quantifying phylogenetic similarity

dist.topo(raw.rooted, F84.rooted,method="score")
```

```
##            tree1
## tree2 0.04219896
```

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 as phyDat object

phyDat.aln <- msaConvert(read.aln, type = "phangorn::phyDat")

# Make the NJ tree for the maximum likelihood method
```

```r
aln.dist <- dist.ml(phyDat.aln)
aln.NJ <- NJ(aln.dist)

# Fit tree using Jukes-Cantor substitution model

fitJC <- optim.pml(pml(tree = aln.NJ, data = phyDat.aln), TRUE)
```

```
## optimize edge weights:  -10571.04 --> -10396.64
## optimize edge weights:  -10396.64 --> -10396.64
## optimize topology:  -10396.64 --> -10341.45  NNI moves:  10
## optimize edge weights:  -10341.45 --> -10341.45
## optimize topology:  -10341.45 --> -10341.45  NNI moves:  0
```

```r
# Fit tree using GTR model with gamma-distributed rates

fitGTR <- optim.pml(pml(tree = aln.NJ, data = phyDat.aln), TRUE,
                    model = "GTR", optGamma = TRUE, optInv = TRUE,
                    rearrangement = "NNI", control = pml.control(trace = 0))
```

```
## only one rate class, ignored optGamma
```

```r
# Perform model selection with either an ANOVA test or with AIC
anova(fitJC, fitGTR)
```

```
## Likelihood Ratio Test Table
##   Log lik. Df Df change Diff log lik. Pr(>|Chi|)
## 1 -10341.5 77
## 2  -9790.4 86         9        1102.1  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
AIC(fitJC)
```

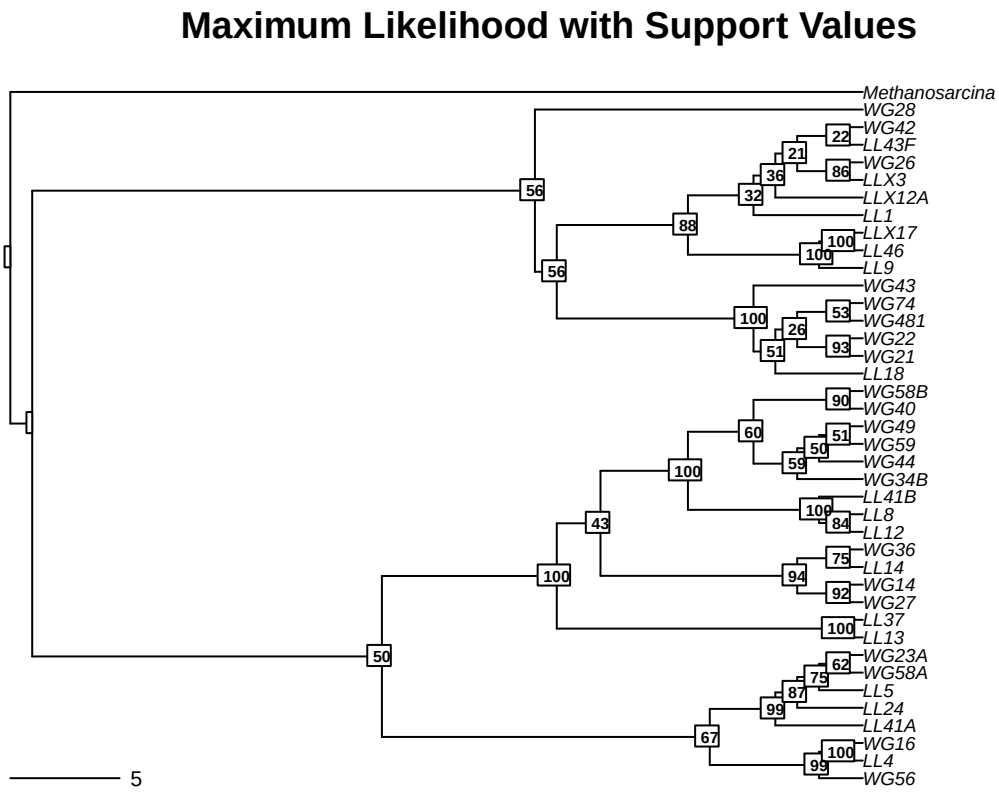
```
## [1] 20836.9
```

```r
AIC(fitGTR)
```

```
## [1] 19752.84
```

```r
###-----bootstrapping

# Read the bootstrap tree
ml.bootstrap <- read.tree("./data/ml_tree/RAxML_bipartitions.T1")

# Set plot margins
par(mar = c(1, 1, 2, 1) + 0.1)

# Plot the tree with bootstrap support values
plot.phylo(ml.bootstrap,
           type = "phylogram",
```

```
            direction = "right",
            show.tip.label = TRUE,
            use.edge.length = FALSE,
            cex = 0.6,
            main = "Maximum Likelihood with Support Values")

# Add a scale bar
add.scale.bar(cex = 0.7)

# Add bootstrap values to the nodes
nodelabels(ml.bootstrap$node.label,
            cex = 0.5,
            font = 2,
            bg = "white",
            frame = "r")
```



Question 4:

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? Why or why not?

Answer 4a:ML makes multiple trees as opposed to a single tree made with neighborhood - joining. ***Answer 4b***: Bootstrapping is a way to see how consistenyl we get the same trees. This is helpful since we're looking for statistical tests. ***Answer 4c***: If you have high values there is strong support for the tree. Low values mean uncertainty on what the trees look like in terms of splits. ***Answer 4d***: Low bootstrapping values have low support ***Answer 4e***: These are not to be trusted since they have a higher likelihood of being incorrect.

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 Growth Rate Data

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

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

B. Trait Manipulations

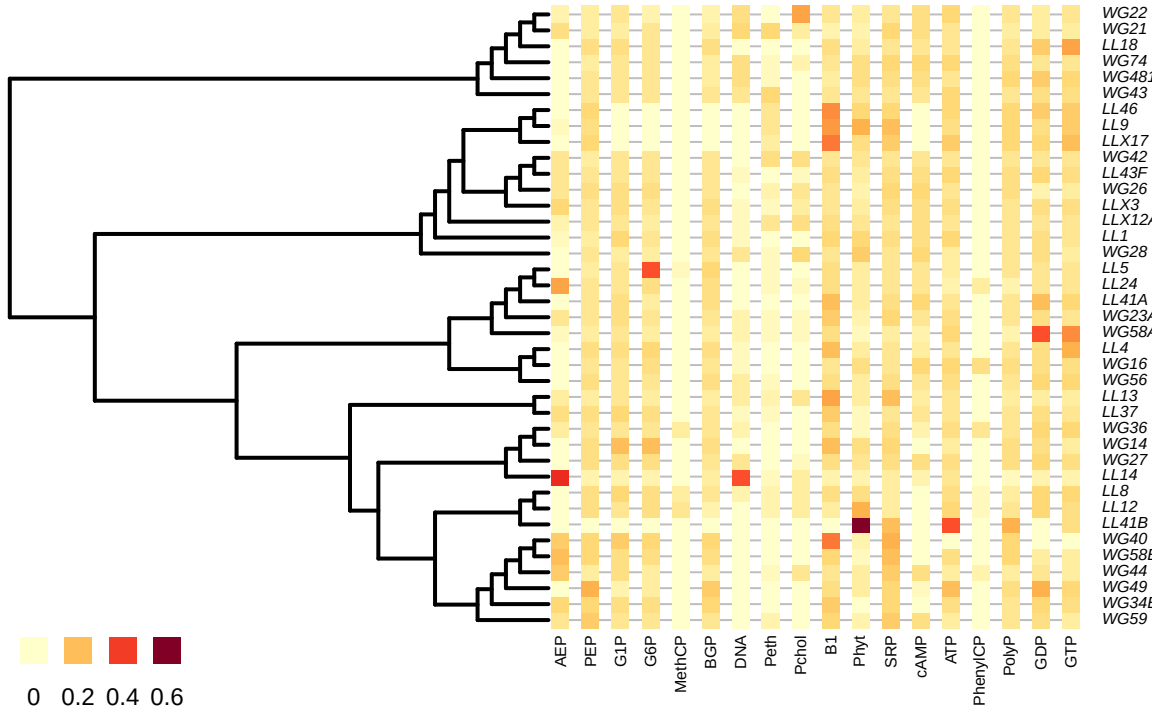
In the R code chunk below, do the following:
1. calculate the maximum growth rate (μ_{max}) of each isolate across all phosphorus types,
2. create a function that calculates niche breadth (nb), and
3. use this function to calculate nb for each isolate.

```
#calcualte max growth rate
umax<-(apply(p.growth,1,max))

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

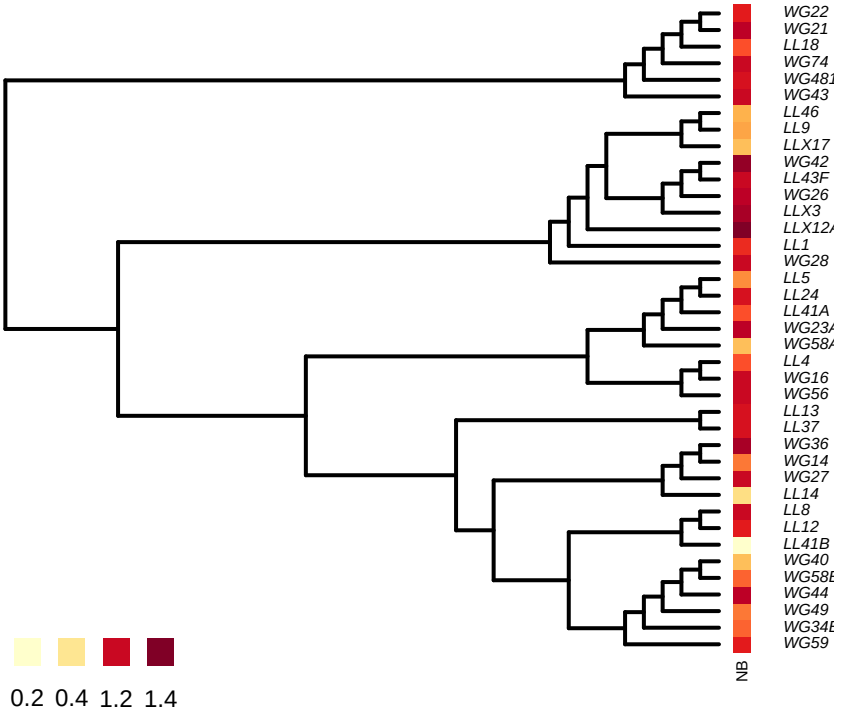
In the R code chunk below, do the following:
1. pick your favorite substitution model and make a Neighbor Joining tree,
2. define your outgroup and root the tree, and
3. remove the outgroup branch.

```
# Generate Neighbor Joining Tree
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")

# Plot to look at our tree


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

# First, Correct for Zero Branch-Lengths on Our Tree
nj.plot <- nj.rooted
nj.plot$edge.length <- nj.plot$edge.length + 10^-1

# Map Phosphorus Traits {adephylo}

# Set plot margins
par(mar = c(1, 1, 1, 1) + 0.1)

# Map growth rates onto the tree


x <- phylo4d(nj.plot, 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)
```



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).

```
# Niche Breadth Map

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 = 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 5:

a) Develop 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 5a: I hypothesize that if a species has a large niche breadth their are likely to have a moderate fitness with most resources (generalist), however if a species has a small niche breadth they are likely to have high fitness when consuming specific resources but comparatively low fitness when consuming other resources. ***Answer 5b***: I expect a negative relationship between maximum growth and niche breath if this hypothesis is true.

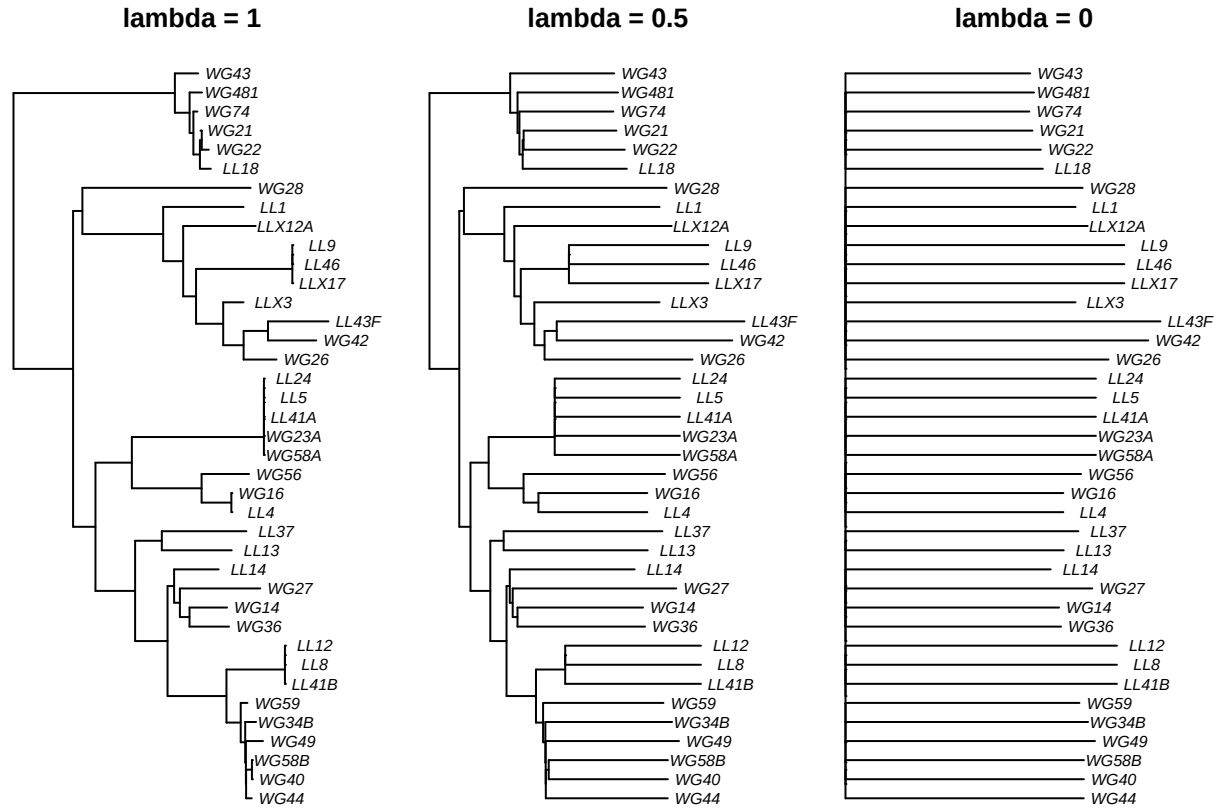
6) HYPOTHESIS TESTING

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.

```
library(geiger)
nj.lambda.5 <- geiger:::rescale.phylo(nj.rooted, model = "lambda", lambda = 0.5)
nj.lambda.0 <- geiger:::rescale.phylo(nj.rooted, model = "lambda", 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 = "lambda = 0.5", cex = 0.7, adj = 0.5)
plot(nj.lambda.0, 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")
```

```
## GEIGER-fitted comparative model of continuous data
##  fitted 'lambda' model parameters:
##  lambda = 0.006976
##  sigsq = 0.108060
##  z0 = 0.657697
##
##  model summary:
##  log-likelihood = 21.503414
##  AIC = -37.006827
##  AICc = -36.321113
##  free parameters = 3
```

```
##
## Convergence diagnostics:
##  optimization iterations = 100
##  failed iterations = 49
##  number of iterations with same best fit = NA
##  frequency of best fit = NA
##
##  object summary:
##  'lik' -- likelihood function
##  'bnd' -- bounds for likelihood search
##  'res' -- optimization iteration summary
##  'opt' -- maximum likelihood parameter estimates
```

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

```
## GEIGER-fitted comparative model of continuous data
##  fitted 'lambda' model parameters:
##  lambda = 0.000000
##  sigsq = 0.108048
##  z0 = 0.656477
##
##  model summary:
##  log-likelihood = 21.502505
##  AIC = -37.005010
##  AICc = -36.319295
##  free parameters = 3
##
## Convergence diagnostics:
##  optimization iterations = 100
##  failed iterations = 0
##  number of iterations with same best fit = 88
##  frequency of best fit = 0.880
##
##  object summary:
##  'lik' -- likelihood function
##  'bnd' -- bounds for likelihood search
##  'res' -- optimization iteration summary
##  'opt' -- maximum likelihood parameter estimates
```

```
#Compare Pagel's lambda score with likelihood ratio test
#lambda = 0, no phylogenetic signal

phylosig(nj.rooted, nb, method = "lambda", test = TRUE)
```

```
##
## Phylogenetic signal lambda : 0.00699105
## logL(lambda) : 21.5034
## LR(lambda=0) : 0.00181763
## P-value (based on LR test) : 0.965994
```

Question 6: 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?

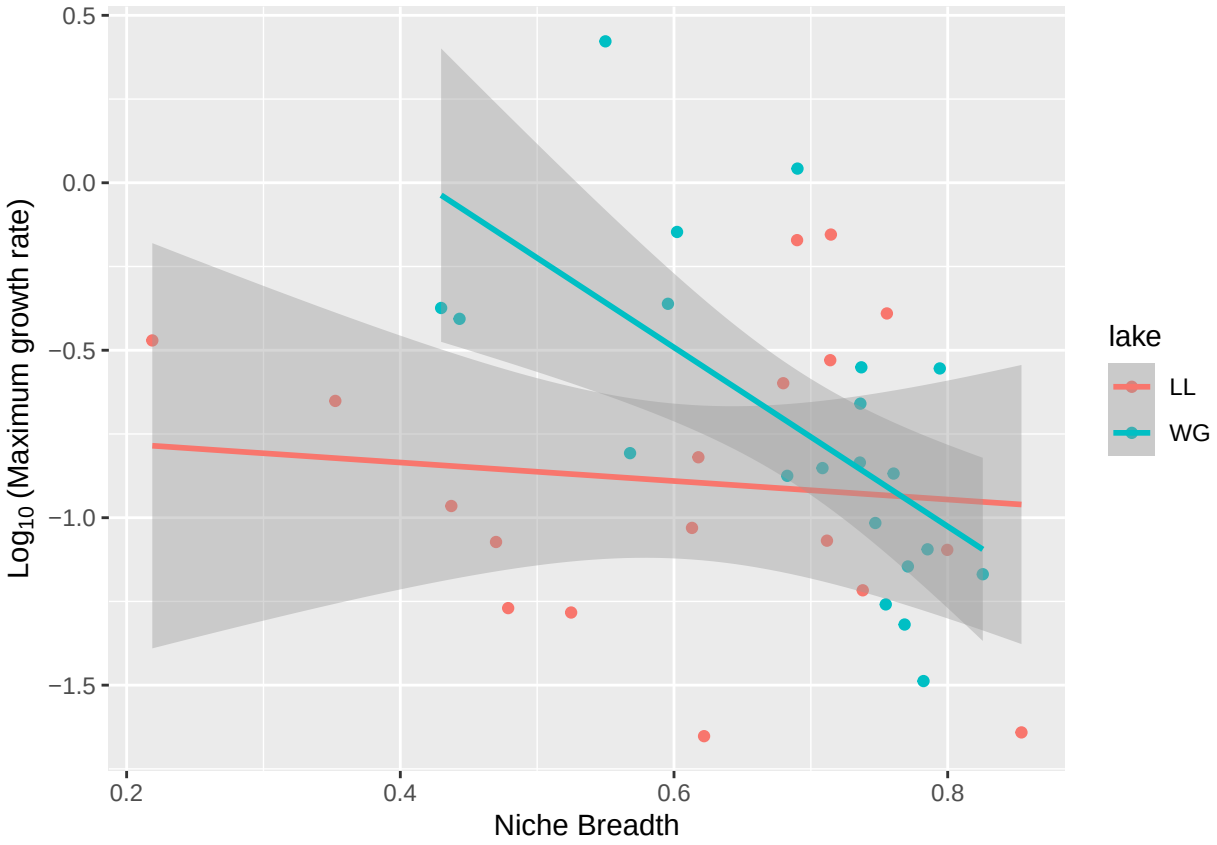
> ***Answer 6a***:The fit shows lambda ~0.007 for the untransformed tree (which is very close to 0) ***Answer 6b***:The AIC values for the tree with lambda ~0.007 vs. lambda=0 are nearly identical – indicating no significant difference. ***Answer 6c***: There's no strong phylogenetic signal for niche breadth (the likelihood ratio test is nonsignificant)

7) PHYLOGENETIC REGRESSION

Question 7: In the R code chunk below, do the following:
1. Clean the resource use dataset to perform a linear regression to test for differences in maximum growth rate by niche breadth and lake environment. 2. Fit a linear model to the trait dataset, examining the relationship between maximum growth rate by niche breadth and lake environment, 2. Fit a phylogenetic regression to the trait dataset, taking into account the bacterial phylogeny

```
#Using the niche breadth data, create a column that indicates the lake origin of each strain



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

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

#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)"))
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
#Simple linear regression

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

```
##
## Call:
## lm(formula = log10(umax) ~ NB * lake, data = nb.lake)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7557 -0.3108 -0.1077  0.3102  0.7800
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.7247     0.3852  -1.882   0.0682 .
## NB           -0.2763     0.6097  -0.453   0.6533
## lakeWG        1.8364     0.6909   2.658   0.0118 *
## NB:lakeWG    -2.3958     1.0234  -2.341   0.0251 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.418 on 35 degrees of freedom
## Multiple R-squared:  0.2595, Adjusted R-squared:  0.196
## F-statistic: 4.089 on 3 and 35 DF,  p-value: 0.01371
```

```r
AIC(fit.lm)
```

```
## [1] 48.413
```

```r
#  Run a phylogeny-corrected regression with no bootstrap replicates

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

```
##
## Call:
## phylolm(formula = log10(umax) ~ NB * lake, data = nb.lake, phy = nj.rooted,
##     model = "lambda", boot = 0)
##
##    AIC logLik
##  41.08 -14.54
##
## Raw residuals:
##      Min       1Q   Median       3Q      Max
## -0.75804 -0.18999 -0.07425  0.32496  0.95857
##
## Mean tip height: 0.1814501
## Parameter estimate(s) using ML:
## lambda : 0.4861372
## sigma2: 0.9184437
##
## Coefficients:
##              Estimate    StdErr t.value p.value
## (Intercept) -0.891268  0.370036 -2.4086 0.02142 *
## NB          -0.004805  0.521303 -0.0092 0.99270
## lakeWG       1.438930  0.577231  2.4928 0.01755 *
## NB:lakeWG   -1.966388  0.848702 -2.3169 0.02648 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-squared: 0.1935    Adjusted R-squared: 0.1243
##
## Note: p-values and R-squared are conditional on lambda=0.4861372.
```

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 7a: Simply put because we can incorrectly infer about species resource consumption without accounting for phylogeny. Similarity in niche can be because of convergence or because of shared ancestry, this must be distentangled. ***Answer 7b***: It includes a correlation structure derived from the tree, whereas standard linear regression treats each species as fully independent. ***Answer 7c***: In this example, we see some difference in the intercept and slope, but overall it

doesn't drastically change the story. The AIC for fit.plm is 41.08 which is lowest value. Hence the phylogenetic model fits the best (comparatively). ***Answer 7d***: If one small clade drives the entire correlation, factoring in phylogeny can kill the correlation if the rest of the tree doesn't share that pattern.

7) SYNTHESIS

Work with members of your Team Project to obtain reference sequences for 10 or more 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?

SUBMITTING YOUR ASSIGNMENT

Use Knitr to create a PDF of your completed `8.PhyloTraits_Worksheet.Rmd` document, push it to GitHub, and create a pull request. Please make sure your updated repo include both the pdf and RMarkdown files. Unless otherwise noted, this assignment is due on **Wednesday, February 26th, 2025 at 12:00 PM (noon)**.