Week 08: Phylogenies!

Due March 13th

I. General

Type up all of your work in a text editor.

Before you begin, make a new folder in Tasks called Task_08, and save an empty file named task08.r in that folder.

You must also make a separate "Project" folder within Tasks. Inside your Project folder, make "code", "data", and "text" subfolders. By Friday, you should have some data in the data folder, some code to read the data in to `R` in the code folder, and a document with a *single sentence* describing your hypothesis.

When you're **done** with this assignment, turn it in by (1) saving your text document, (2) opening your Terminal or GitBash, (3) navigating to the appropriate directory using `cd` and (4) typing:

`git add -A` (enter)
`git commit -m "Task 08"` (enter)
`git push -u origin master` (enter)

The main goals this week are two fold: first, you should get comfortable reading and interpretting phylogenies in general, and second, you should get comfortable manipulating phylogenies in `R`. A lot of you are going to need to be able to read and manipulate phylogenies in `R` for your final project, and *all* of you will need to *deeply and completely* understand how to read phylogenies for the course. So while this assignment is a bit short, you should endeavor to go above and beyond and manipulate the trees so that you convince yourself you can fully grok the diagrams. This week, I need just your `R` code and the single plot generated in question 10.

II. Project

March 25th an almost-complete dataset and an analysis complete enough to produce a plot pertinent to your hypothesis is due. About a week and a half after that is when you'll need to have a fully realized draft of your paper. So basically between now and the 25th you need to focus on getting enough data, and ensuring you have code that can (1) read your data in, (2) do some simple analysis, and (3) produce a graph that tests your hypothesis. If you get all of that up-and-running by the 25th you'll be in great shape.

III. Your Code!

```
# This week, we're going to look at evolutionary trees in R. Evolutionary
    trees are also known as phylogenies, and they describe the pattern of
    relationships between lineages. So like a pedigree, but for species
    isntead of individuals.

# First, you'll need to load the phytools and ape libraries. You should have
    them installed, but if not you can just install them now.
```

```

# Now we're going to input a tree, and plot it.
# We're going to hand-input this tree. We'll won't enter trees this way
   after this week, but it's worth seeing how trees are stored.
text.string<-
    "(((((((cow, pig),whale),(bat,(lemur,human))),(robin,iguana)),coelacanth
       ),(gold_fish, trout)),shark);"
vert.tree<-read.tree(text=text.string)
plot(vert.tree, edge.width=2)

# Reading these trees can be a bit tricky. The number of nodes that separate
    two lineages reflects how distantly related they are.
nodelabels(frame="circle", bg='white', cex=1)

# So, you can determine how distant two lineages are looking at when the
   most recent node (their most recent common ancestor) is. So for instance,
    if you go from humans to lemurs, you hit node number 21. So both humans
   and lemurs are descended from node 21, and no other organisms are. Whales
    and humans are both descended from node 17, but iguanas are not, so
   whales and humans are closer than either are to iguanas.

# QUESTION 1: What's more closely related to the gold fish, a shark or a
   human?


# The object created in memory when we simulate or estimate a phylogeny, or
   read one from an input file, is a list of class "phylo".
# Remember, a list is just a customizable object type that can combine
   different objects of different types. For instance, a list might have a
   vector of real numbers (with mode "numeric") as its first element; and
   then a vector of strings (with mode "character") as its second element;
   and so on. Assigning our tree with a special class, "phylo", is just a
   convenient way to tell special functions in R how to treat that object.
# An object of class "phylo" has at least three parts. These are normally
   hidden, for instance, just typing the name of your "phylo" object does
   not give you the structure in memory, as it does for many R objects

# Let's look at the object
vert.tree

# QUESTION 2: Are there branch lengths in this tree?
str(vert.tree)

# Okay, so we're going to dig a bit into the phylo object. We'll use a
   simpler tree to explore this.
tree<-read.tree(text="(((A,B),(C,D)),E);")
plotTree(tree,offset=1)
tiplabels(frame="circle", bg='lightblue', cex=1)
nodelabels(frame="circle", bg='white', cex=1)

```

```
# So the tip called "E" is numbered 5. It's descended from node 6. Likewise,
    tip number 4 is descended from node number 9.
# We can call them using
tree$tip.label

# Note that the FIRST element of tree$tip.label is "A", and that A is listed
    as tip #1 on the tree. That's not a coincidence! That's how the tip
   labels are linked to the edge matrix of the tree. What's the edge matrix?
    Read on!

# If we look into the phylo object's edge component, we can see the
   structure of the phylogeny as a matrix
tree$edge

# Each line on the phylogeny is called an ‘‘edge’’. So each row of tree$edge
    corresponds to one of the lines (edges) of the phylogeny. The first
   number is where the line starts, the second number is where the line ends
   . So the first row shows the first line starts at number 6 and goes to
   number 7.

# Now we'll use a real phylogeny of Anolis lizards (which are super cute)
AnolisTree <- force.ultrametric(read.tree("https://jonsmitchell.com/data/
   anolis.tre"))

# Unlike the previous tree, this one has lengths associated with each edge.
   So edge lengths.
par(las=1)
hist(AnolisTree$edge.length, col='black', border='white', main="", xlab="
   edge lengths for the Anolis tree", ylim=c(0, 50), xlim=c(0, 6))


tipEdges <- which(AnolisTree$edge[,2] <= Ntip(AnolisTree))
Lengths <- AnolisTree$edge.length
names(Lengths) <- AnolisTree$tip.label
names(Lengths)[which(Lengths == min(Lengths))]

plot(AnolisTree, cex=0.25)
Labs <- sapply(AnolisTree$edge.length, round, digits=2)
edgelabels(text=Labs, cex=0.25)

# Each edge has a length, and they're in order. So the first value of edge
   length is the length of the edge defined by the first row of the edge
   matrix.
# Don't gloss over the line above this one! It's crucial to the later
   questions.

# Okay, so now I want you to do ?plot.phylo and look through the options.
   Then write code that:

# QUESTION 3: A tree with no tip labels
# QUESTION 4: A tree that is plotted as a circle, instead of facing right or left
# QUESTION 5: A tree with the tips colored red instead of black

# Okay, now you're going to use the which() function in combination
# QUESTION 6 - 8: Find 1) which living, named species has the shortest edge length (NOT necessarily the shortest overall length!), then 2) drop that tip from the tree, then 3) plot the resulting tree.

# Let's look at how fast species appeared in this phylogeny. We'll do this by making a lineage-through-time (ltt) plot. This shows the number of observed lineages alive at any given time.
ltt(AnolisTree)
abline(0, 1, lwd=2, col='red', lty=2)

# What do you notice about that line? Does it ever go down? Why not? Is the slope always the same? What does the slope of this curve tell you about how these lizards?

# Finally, QUESTION 10: Use the function fit.bd() to calculate the rate new species form (b) and disappear (d) in Anolis lizards. Set rho = 0.2
```

IV. Extra credit

Download and investigate the package `treebase`. Use that package to obtain phylogenies of ten different groups (e.g., warblers, skinks, and tree frogs could be three example groups). Then calculate the `b` & `d` values as above for each group and plot them against the number of species in each tree.