Fall 2022 Midterm

Daisy Fry-Brumit

2022-10-04

Question 1: What methods and models did I use to create the data and the reference alignment that I have provided you? (10 pts)

1A. Which of the 3 alignment methods available in the R msa() package did I use to make the provided reference alignment file? Be sure to explain your answer, and how you got there (don't just guess at one of the three). (5 pts)

To answer this question, I ran msa() using all 3 possible methods and output each result as a fasta file as seen in the code snippet below.

```
# initial sequence information
inputSeqs = readDNAStringSet('pumpkin-input.fa')

# start out by running msa() with all possible methods
cw_align = msa(inputSeqs, method = "ClustalW")
co_align = msa(inputSeqs, method = "ClustalOmega")
muscle_align = msa(inputSeqs, method = "Muscle")

# generate fasta files of the alignments
cw_align_align = msaConvert(cw_align, "bios2mds::align")
export.fasta(cw_align_align, outfile= 'cw_aln.fa')

co_align_align = msaConvert(co_align, "bios2mds::align")
export.fasta(co_align_align, outfile= 'co_aln.fa')

muscle_align_align = msaConvert(muscle_align, "bios2mds::align")
export.fasta(muscle_align_align, outfile= 'muscle_aln.fa')
```

After generating the .fa files, I used VerAlign online to compare my output files with *pumpkin-refaln.fa* as a reference, with results:

Clustal-W SP = 1.00 | CS = 0.99 | avg SPdist = 1.00

Clustal-O SP = 1.00 | CS = 1.00 | avg SPdist = 1.00

Muscle SP = 0.83 | CS = 0.60 | avg SPdist = 0.90

Although Clustal W generated a near-perfect match, Clustal-Omega generated an extact match to the rerence alignment. Thus, **the reference is a CLustal-Omega-generated alignment**.

1B. To create the input data set I provided you, I took real data and simulated additional mutations that would mimic the real evolutionary history of pumpkins. What mutation model did I use, and how did you come to that conclusion? Also, please provide a brief description of this model in terms of its parameters and assumptions. (5 pts)

To start I used modelTest() to return scores associated with each possible mutation model. To cover my bases, I did not subset the models used like we have previously in class.

```
# convert aligned sequences to phangorn friendly format
forPhang = msaConvert(co_align, type = "phangorn::phyDat")

# test all models on alignment
model_test = modelTest(forPhang)
```

Because I ran so many tests, I'm not going to print out the full table of results and eyeball the BIC values. I'll filter for the lowest BIC value instead:

```
model_test[which(model_test$BIC == min(model_test$BIC)), ]
```

```
##   Model df    logLik      AIC       AICw     AICc      AICcw      BIC
## 1    JC 55 -36188.77 72487.55 0.01152535 72491.81 0.01565719 72779.77
```

Based on the filter applied, it seems like the **Jukes Cantor model was used to generate mutations**. The JC model is a simple, one-parameter model that assumes

- base frequencies are equal
- rates of substitution among our bases are also equal (regardless of substitution type)

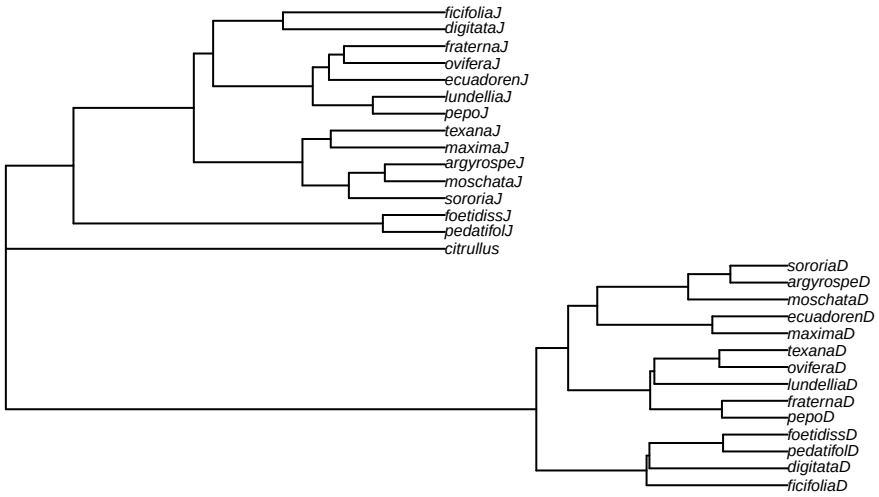
Question 2: Which set of orthologous genes best captures the evolutionary relationships shown in the reference tree that I provided? (10 pts)

To visualize the clades in the dataset and differentiate the orthologous groups, I have to build a tree. To do this, I started by using the JC69 model to get distance metrics, and then use those distance metrics to build and compare UPGMA and Neighbor-Joining tree methods.

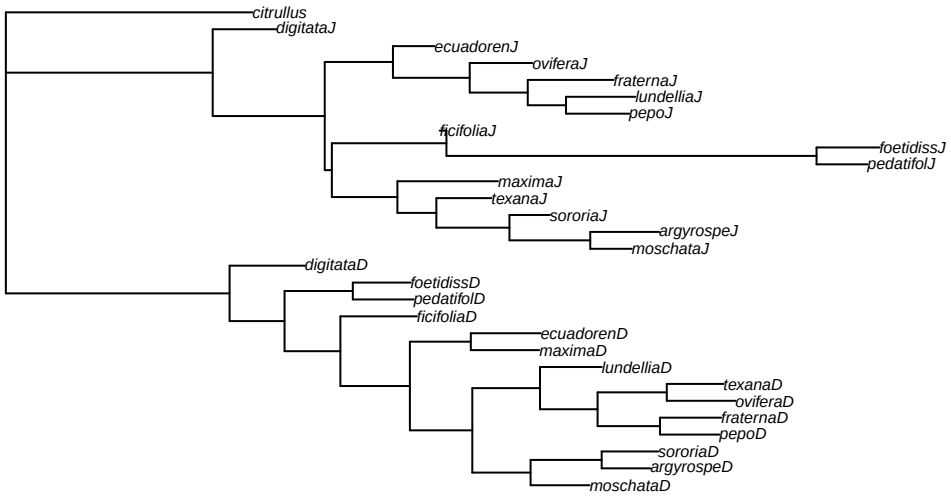
```
# convert forPhang data into dist.dna useable format, then generate pairwise dist metrics
#names(forPhang) = gsub("D", "", names(forPhang))
#names(forPhang) = gsub("J", "", names(forPhang))
dna = as.DNAbin(forPhang)
D = dist.dna(dna, model='JC69')

# generate nj and upgma trees
upgma_tree = upgma(D)
nj_tree = nj(D)

# root using citrullus (watermelon) as outgroup
upgma_tree_root = root(phy=upgma_tree, outgroup="citrullus")
nj_tree_root = root(phy=nj_tree, outgroup="citrullus")

# input ref tree
refTree = ape::read.tree('pumpkin-refTree.nwk')
```

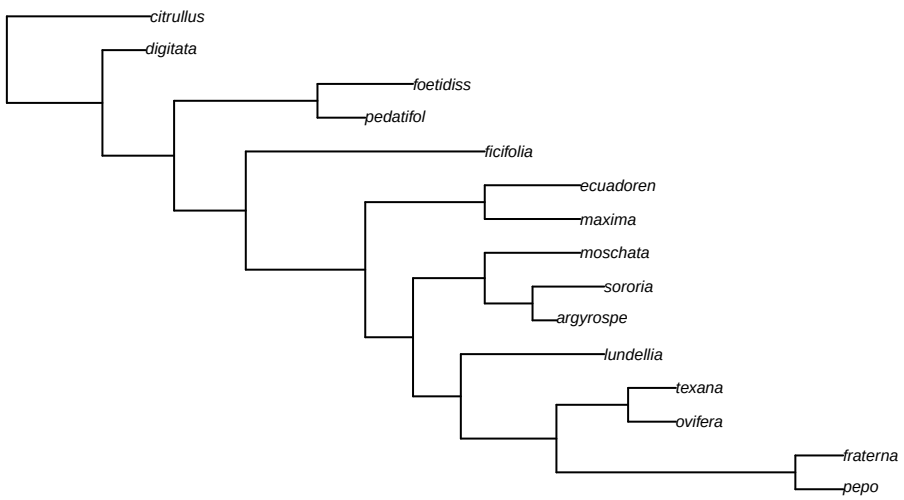
The resulting trees are as follows:

UPGMA Rooted Tree



NJ Rooted Tree





To me, it looks like the "D" orthologs generated by the Neighbor Joining method are the closest matches to the reference tree, but I want to use distance metrics to better evaluate the groups. To do this, I will compare the D and J groups from each tree building method to the reference tree (so 4 total comparisons using treedist). The results can be seen in the following tables:

Table 1: D Group Comparison (NJ method)

	x
symmetric.difference	0.0000000
branch.score.difference	0.7377075
path.difference	0.0000000
quadratic.path.difference	5.1148061

Table 2: D Group Comparison (UPGMA method)

	x
symmetric.difference	8.0000000
branch.score.difference	0.8465216
path.difference	12.9614814
quadratic.path.difference	5.3590267

Table 3: J Group Comparison (NJ method)

	x
symmetric.difference	16.0000000
branch.score.difference	0.9430923
path.difference	20.4450483
quadratic.path.difference	4.7939323

Table 4: J Group Comparison (UPGMA method)

	x
symmetric.difference	20.0000000
branch.score.difference	0.9465929
path.difference	21.8632111
quadratic.path.difference	4.8352514

I can see from the trees and verify from the comparison tables that the "D" group generated by the Neighborhood Joining method has no unique clades when compared to the reference (symmetric.difference) and there is also no path difference. This leads me to believe that **the orthologous group containing "D" subunits on the chloroplast gene *psb* best captures the relationships provided by the reference.**

Note this part of the code took me a while to figure out (in terms of comparing each group separately to the reference) and so the code generating the displayed tables is longer and less efficient than I would care for. Thus, I'll upload the code for that separately so it doesn't ruin the flow of the current file.

```
# start with bootstrapped ML tree (using rooted "D" subunit group generated by NJ method)
nj_boot = boot.phylo(nj_tree_root_NOD, dna_NOD, function(x) root(nj(dist.dna(x, model="JC69")),1),trees=
```

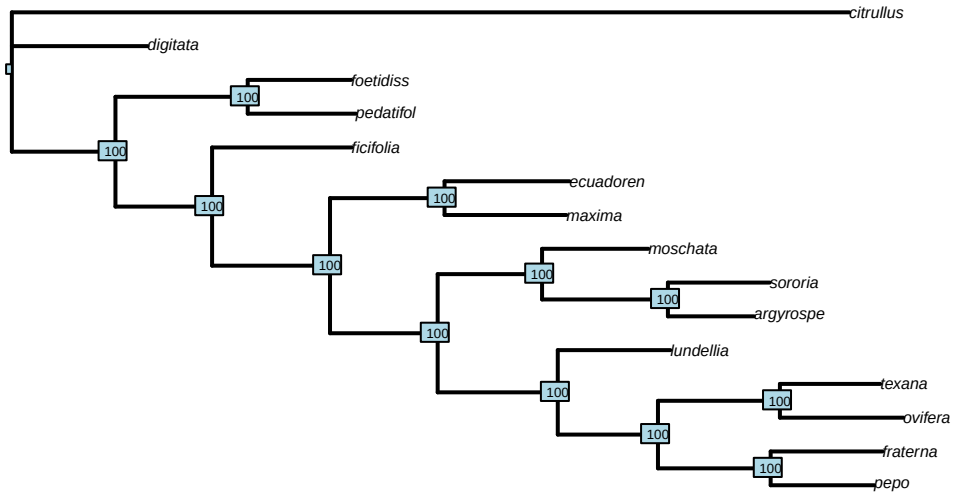
```
## Running bootstraps:       100 / 100
## Calculating bootstrap values... done.
```

```
upgma_boot = boot.phylo(upgma_tree_root_NOD, dna_NOD, function(x) root(nj(dist.dna(x, model="JC69")),1)
                        )
```

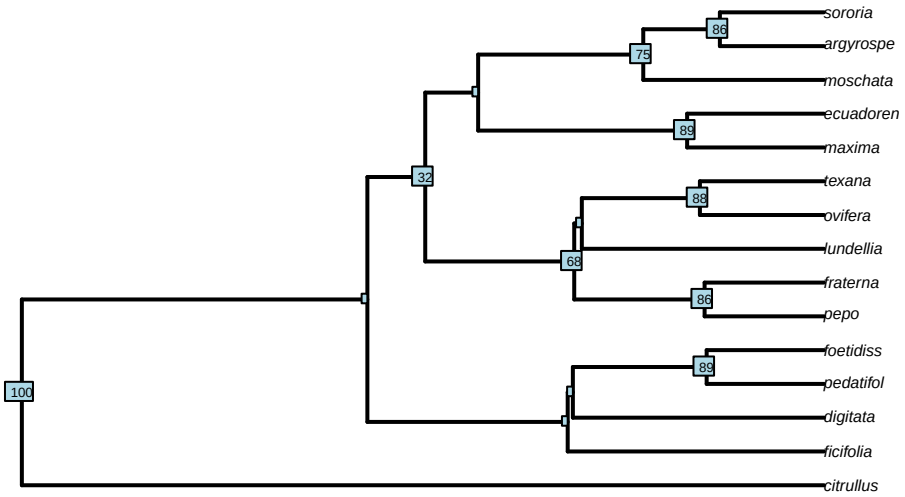
```
## Running bootstraps:       100 / 100
## Calculating bootstrap values... done.
```

```
plot(nj_tree_root_NOD, cex=0.5, edge.width=2, main="Bootstrapped Tree: D subunits, NJ generated")
nodelabels(nj_boot$BP, cex=0.4)
```

Bootstrapped Tree: D subunits, NJ generated




```
plot(upgma_tree_root_NOD, cex=0.5, edge.width=2, main="Bootstrapped Tree: D subunits, UPGMA generated")
nodelabels(upgma_boot$BP, cex=0.4)
```




```
#
```

Fortunately, the NJ generated tree has perfect bootstrapping values and lends a good deal of security when inferring domestication events. Since the last step, I've pretty much known that the NJ method is preferable to the UPGMA method given our reference and tree comparisons, but it never hurts to compile extra evidence, especially when the runtime for doing so is short. **Going forward, I'll use the NJ tree output where appropriate.**

Question 3: How many independent domestication events can you infer under the most likely transition cost model, and what is the most likely model? (10 pts)

To run RAxML for the appropriate orthologous group, I used the appropriate subset of data, converted it to phylip format, and executed the job script from the HPC cluster. I ran 3 jobs on my data and checked for convergence. All 3 Final ML Optimization Likelihood values were idenitcal at -18836.808882. I downloaded the results from one of the runs to use for subsequent steps.

To determine the most likely transition model, I input my RAxML tree results and fit the "all results different" option with `ace` as shown in the Ancestral State Reconstruction practice on Canvas. Then I did the same thing but with the equal states option and compared output using a chi-squared test, which yielded a value of 0.7962596.

```
# output phylip
write.phylip(align_D, "pumpkins_D.phy")
```

```
# Set up RAxML tree and generate "wild" labels and species names
mlTree = read.tree('RAxML/RAxML_bestTree.pumpkinsBoot1')
TreeID = c("digitata","foetidiss","pedatifol","ficifolia","ecuadoren","maxima","lundellia",
           "pepo","fraterna","ovifera","texana","argyrospe","sororia","moschata","citrullus")
wild = c(2,2,2,1,2,1,2,1,2,1,2,1,2,1,1)
names(wild) = TreeID

# find the most likely transition model (discrete, as used in class practice)
fitARD = ace(x=wild, phy=mlTree, type="discrete", model="ARD")
```

```
## Warning in sqrt(diag(solve(h))): NaNs produced
```

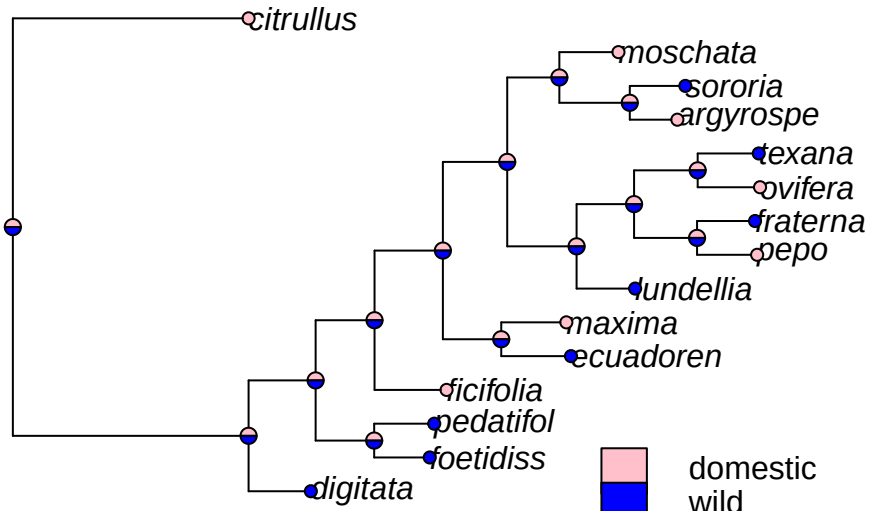
```
fitER = ace(x=wild, phy=mlTree, type="discrete", model="ER")
x = 2*(fitARD$loglik - fitER$loglik)
pchisq(x, df=1, lower.tail = FALSE) # 0.7962496
```

```
## [1] 0.7962496
```

```
# build a pie matrix using ARD results
pie_matrix = matrix(fitER$lik.anc, ncol=2)
rownames(pie_matrix) = seq(16,29)
colnames(pie_matrix) = c("domestic", "wild")
```

The chi-squared value of >0.7 suggests that there isn't a significant difference between ER and ARD models, so I assume **the equal rates model is the best fit for the data** and produced the tree below:

RAxML Output with Equal Rated Fitted Transition Costs

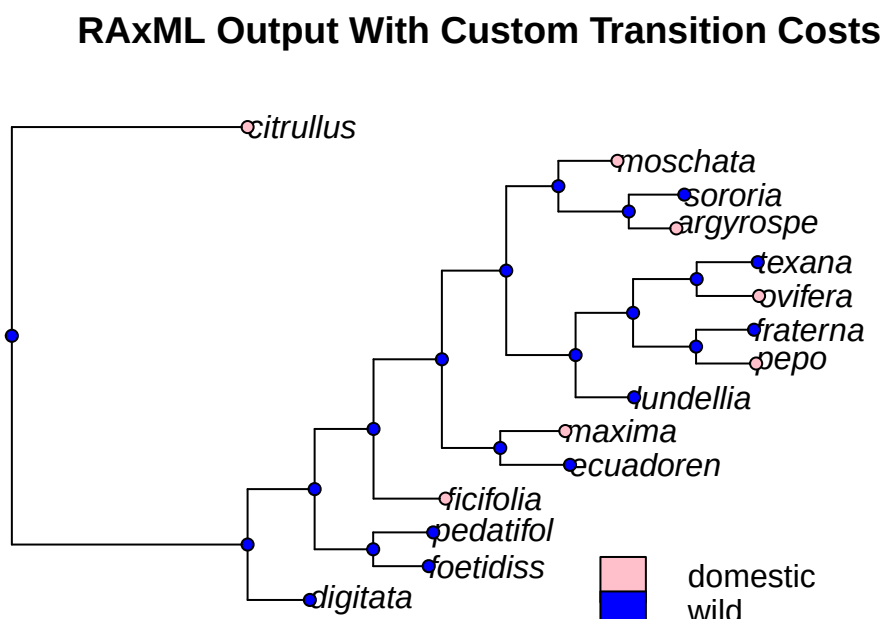


With this tree, I cannot get a clear read on how many independent domestication events occurred because there's a 50/50 chance of each ancestral state being domesticated or wild, leaving me with little basis for saying *where* domestication occurred.

Question 4: If you assume domestication is irreversible, then how many separate times did domestication occur in these species? (10 pts)

To answer this question, I simply created a custom transition probability matrix Q with no chance of redomestication and printed the corresponding tree:

```
Q = matrix(data=c(0,0,1,0), nrow=2, dimnames=list(c(1,2), c(1,2)))

mp_custom = asr_max_parsimony(tree=mlTree, tip_states=wild, Nstates=2, transition_costs=Q)

### Plot the new results
pie_matrix2 = matrix(mp_custom$ancestral_likelihoods, ncol=2)
rownames(pie_matrix2) = seq(16,29)
colnames(pie_matrix2) = c("domestic", "wild")
plot(mlTree, main="RAxML Output With Custom Transition Costs")
tiplabels(pie = to.matrix(wild, sort(unique(wild))), piecol = c("pink", "blue"), cex = 0.3)
nodelabels(node = as.numeric(rownames(pie_matrix2)), pie = pie_matrix2, piecol = c("pink", "blue"), cex=
add.simmap.legend(leg=c("domestic", "wild"), colors=c("pink", "blue"), prompt=FALSE, x=1, y=1.6)
```



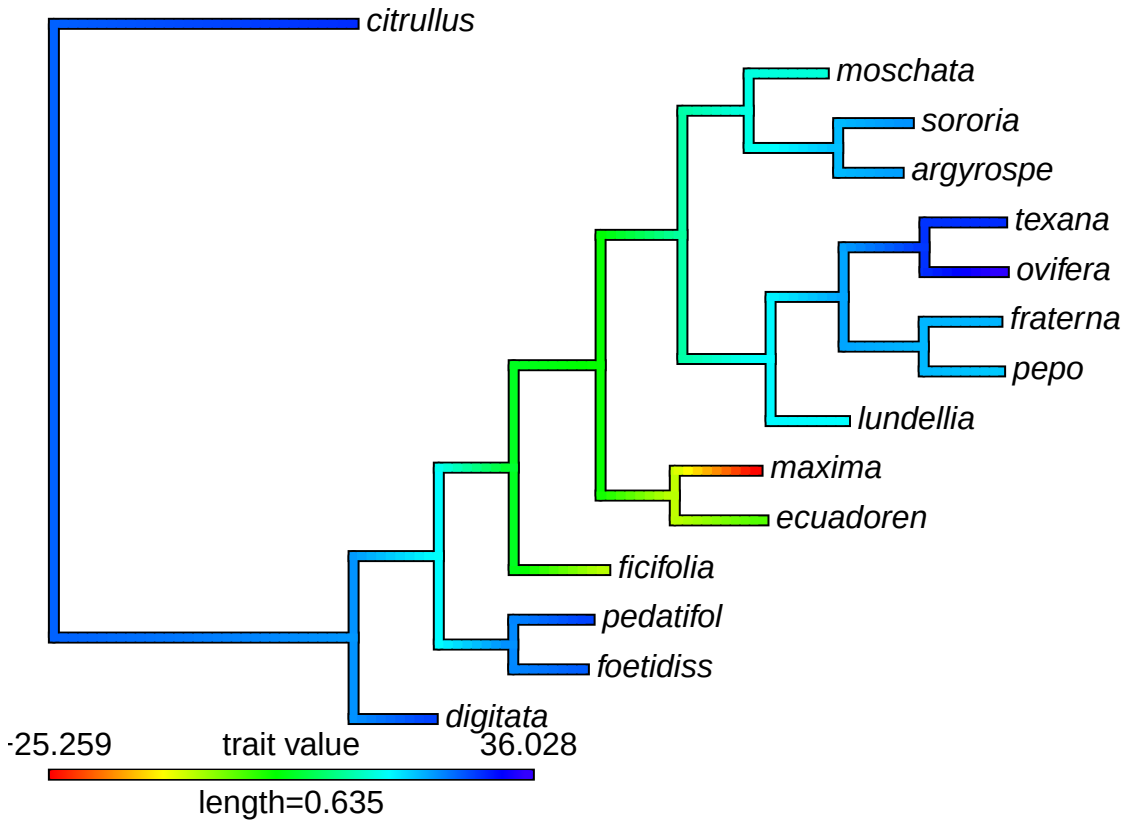
Assuming redomestication is not possible, we see **7 independent domestication events**, one for each terminal node/domestic species represented in the tree.

Question 5: Where was C. ficifolia (the fig-leaf gourd) most likely from? (10 pts)

When I compare the layout of the RAxML generated tree (provided in both plots of Question 4) and the NJ generated tree (provided with bootstrapping values in Question 3), the common theme is that *C. digitata*, *C. foetidissima*, and *C. pedatifolia* all cluster up together just before *C. ficifolia*, all of which are wild and from NW Mexico. In both tree outputs, I can see that *C. ficifolia*'s next closest relatives are suggested to be *C. maxima* and *C. equadorensis*, which are from Argentina/Uruguay and Ecuador, respectively. So off the bat here I have reason to think that multiple tree building methods, maximum likelihood models, and bootstrapping methods all hint at *C. ficifolia* coming from some the region between NW Mexico and Northern/Central South America. It's a large area but I think it narrows down from other possible locations presented in the trees.

To further explore this hunch and narrow down my range to a single country, I'll approach Ancestral State Reconstruction again, this time with latitude as the continuous state of interest. I will use the same RAxML built tree produced by the cluster for my analysis.

```
# assign variable to names, sync with tip labels
lat = meta$Latitude
names(lat) = meta$TreeID
m = match(mlTree$tip.label, names(lat))
lat = lat[m] # lat is now reordered so the species appear as the will on the tree

# print a continuous map with contMap
contMap(mlTree, lat)
```

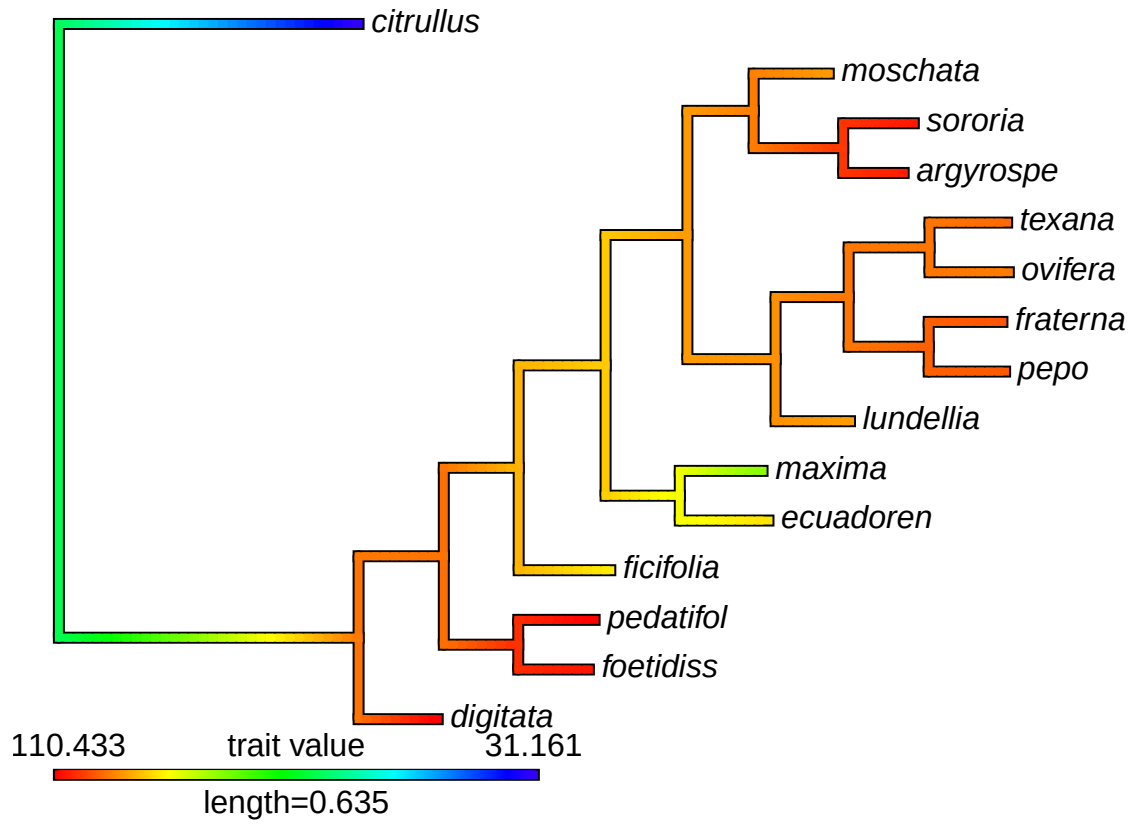


```
# want rough midpoint lat value
((36.028 - -25.259)/2) + -25.259 # 5.3845
```

```
## [1] 5.3845
```

The plot above implements a heat map where anticipated states (here: latitude) are color coded. This employs as continuous maximum likelihood analysis, so at the very least we know that the results are statistically explainable. With this in mind, ML states that *C. ficifolia*'s most likely latitudinal state is around the midpoint of our scale, suggesting a value roughly around 5.3845, which corresponds to Northern South America. To distinguish between Colombia and Guyana, I'll execute the same process for determining the likely longitude.

```
long = meta$Longitude
names(long) = meta$TreeID
m = match(mlTree$tip.label, names(long))
contMap(mlTree, long)
```




```
# want rough 1st quartile long value
((31.161- -110.433)*0.25) + -110.433 # -75.0345
```

```
## [1] -75.0345
```

Estimating a most likely value for longitude here is less simplistic than simply eyeballing a midpoint. I did consult some forums on how to return tip values from the contMap object, but was met with some confusing

advice and so I've decided to stick to the scope of skills taught in our class tutorial. That is to say: I'm going to eyeball it again. The orange-yellow color shown at *C. ficifolia* seems to be roughly 1/4 of the way down our scale, which yields a value of -75.0345.

Coordinates 5.3845,-75.0345 place the location of *C. ficifolia* at Colombia

Appendix

Code from question 2 (comparing various tree builds)

D Groups names(forPhang) = gsub("D", " ", names(forPhang))

forPhang_NOD = forPhang[-c(16:29)]

dna_NOD = as.DNAbin(forPhang_NOD)

D_NOD = dist.dna(dna_NOD, model='JC69')

upgma_tree_NOD = upgma(D_NOD)

nj_tree_NOD = nj(D_NOD)

upgma_tree_root_NOD = root(phy=upgma_tree_NOD, outgroup="citrullus")

nj_tree_root_NOD = root(phy=nj_tree_NOD, outgroup="citrullus")

treeComp_noD_nj = treedist(nj_tree_root_NOD, refTree) # COMP D FROM NJ

treeComp_noD_upgma = treedist(upgma_tree_root_NOD, refTree) # COMP D FROM UPGMA

J Groups forPhang = msaConvert(co_align, type = "phangorn::phyDat")

names(forPhang) = gsub("J", " ", names(forPhang))

forPhang_NOJ = forPhang[-c(1:14)]

dna_NOJ = as.DNAbin(forPhang_NOJ)

D_NOJ = dist.dna(dna_NOJ, model='JC69')

upgma_tree_NOJ = upgma(D_NOJ)

nj_tree_NOJ = nj(D_NOJ)

upgma_tree_root_NOJ = root(phy=upgma_tree_NOJ, outgroup="citrullus")

nj_tree_root_NOJ = root(phy=nj_tree_NOJ, outgroup="citrullus")

treeComp_noJ_nj = treedist(nj_tree_root_NOJ, refTree) # COMP J FROM NJ

treeComp_noJ_upgma = treedist(upgma_tree_root_NOJ, refTree) # COMP J FROM UPGMA