Final Project

Michelle

5/8/2020

#SYNOPSIS #This project involves a using mitochondrial sequence data from 14 species of PRIMATES. #896 nucleotides were collected by a number of researchers in Japan and 14 species data was selected by Masami Hasegawa # D-loop noncoding region and third positions of adjacent coding sequences was selected to get a site that is close to having no rate variation.

#Objective was to used: #Align and convert the data #Estimating tress from distance matrices using neighbor-joining and UPGMA #Compute maximum Likelihood and bootstrapping

```
setwd("~/phylometh_exercises-")
```

#loading Library

```
library(ape)
library(phangorn)
library(seqinr)
```

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

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

#Alignment and Conversion of data #Interleaved:the function starts to read the sequences after it finds one or more spaces (or tabulations). All characters before the sequences are taken as the taxa names after removing the leading and trailing spaces (so spaces in taxa names are not allowed). #It is assumed that the taxa names are not repeated in the subsequent blocks of nucleotides.

```
mammals <- read.dna("~/phylometh_exercises-/primates.dna", format="interleaved")
mammals_phyDat <- phyDat(mammals, type = "DNA", levels = NULL)

# Subset (first ten)
mammals10 <- subset(mammals_phyDat, 1:10)
mammals10_phyDat <- phyDat(mammals10, type = "DNA", levels = NULL)
```

#Comparing different nucleotide or amino acid substitution models

```
mt <- modelTest(mammals10)
```

```
## [1] "JC+I"
## [1] "JC+G"
## [1] "JC+G+I"
## [1] "F81+I"
## [1] "F81+G"
## [1] "F81+G+I"
## [1] "K80+I"
## [1] "K80+G"
## [1] "K80+G+I"
## [1] "HKY+I"
## [1] "HKY+G"
## [1] "HKY+G+I"
## [1] "SYM+I"
## [1] "SYM+G"
## [1] "SYM+G+I"
## [1] "GTR+I"
## [1] "GTR+G"
## [1] "GTR+G+I"
```

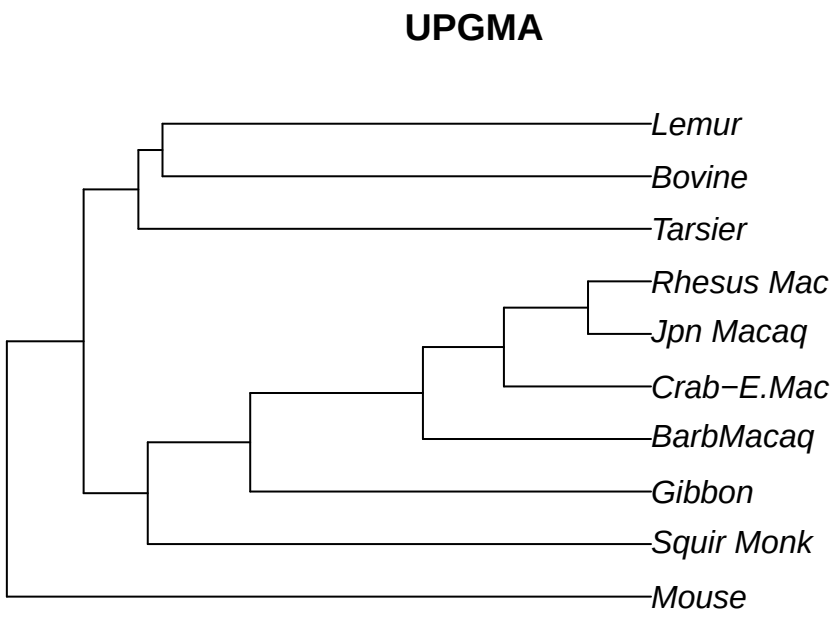
```
print (mt)
```

```
##      Model df    logLik      AIC          AICw     AICc         AICcw      BIC
## 1       JC 17 -2348.727 4731.453 1.015172e-131 4734.313 6.760843e-131 4790.048
## 2     JC+I 18 -2346.692 4729.385 2.856095e-131 4732.596 1.595573e-130 4791.426
## 3     JC+G 18 -2348.426 4732.852 5.045428e-132 4736.063 2.818656e-131 4794.893
## 4   JC+G+I 19 -2346.712 4731.425 1.029722e-131 4735.010 4.772319e-131 4796.913
## 5      F81 20 -2206.835 4453.671  2.119608e-71 4457.652  8.058328e-71 4522.606
## 6    F81+I 21 -2203.862 4449.724  1.525358e-70 4454.124  4.703117e-70 4522.105
## 7    F81+G 21 -2205.054 4452.108  4.629541e-71 4456.508  1.427421e-70 4524.490
## 8  F81+G+I 22 -2203.861 4451.721  5.617673e-71 4456.564  1.388572e-70 4527.550
## 9      K80 18 -2301.019 4638.039 1.955405e-111 4641.250 1.092397e-110 4700.080
## 10   K80+I 19 -2297.749 4633.498 1.893960e-110 4637.082 8.777693e-110 4698.986
## 11   K80+G 19 -2298.851 4635.701 6.293413e-111 4639.286 2.916727e-110 4701.189
## 12 K80+G+I 20 -2297.747 4635.494 6.980091e-111 4639.475 2.653692e-110 4704.429
## 13     HKY 21 -2056.374 4154.748  1.724267e-06 4159.148  5.316410e-06 4227.129
## 14   HKY+I 22 -2048.676 4141.352  1.397503e-03 4146.194  3.454337e-03 4217.181
## 15   HKY+G 22 -2045.236 4134.473  4.357813e-02 4139.315  1.077161e-01 4210.301
## 16 HKY+G+I 23 -2042.963 4131.926  1.556797e-01 4137.234  3.048900e-01 4211.201
## 17     SYM 22 -2195.356 4434.712  2.773590e-67 4439.554  6.855740e-67 4510.541
## 18   SYM+I 23 -2194.096 4434.192  3.598326e-67 4439.499  7.047120e-67 4513.467
## 19   SYM+G 23 -2190.897 4427.794  8.818587e-66 4433.101  1.727071e-65 4507.069
## 20 SYM+G+I 24 -2190.899 4429.797  3.238684e-66 4435.594  4.965990e-66 4512.519
## 21     GTR 25 -2052.104 4154.209  2.257535e-06 4160.519  2.677625e-06 4240.377
## 22   GTR+I 26 -2043.425 4138.850  4.883551e-03 4145.699  4.425923e-03 4228.465
## 23   GTR+G 26 -2039.920 4131.840  1.625575e-01 4138.688  1.473246e-01 4221.455
## 24 GTR+G+I 27 -2037.562 4129.124  6.318996e-01 4136.536  4.321810e-01 4222.186
```

```
dna_dist <- dist.ml(mammals10, model="JC69")
```

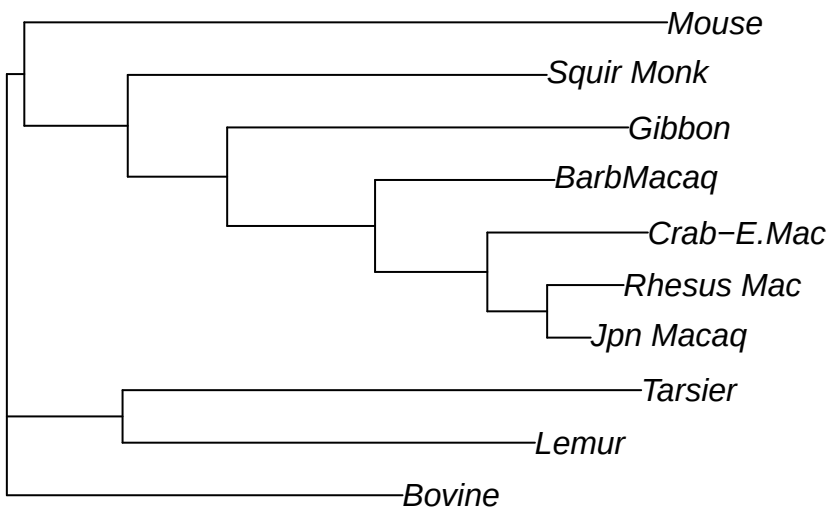
#Estimating tress from distance matrices using neighbor-joining and UPGMA(Unweighted Pair Group Method with #Arithmetic mean) algorithms. #UPGMA is a simple agglomerative hierarchical clustering method

```
mammals_UPGMA <- upgma(dna_dist)
mammals_NJ  <- NJ(dna_dist)
plot(mammals_UPGMA, main="UPGMA")
```



#ploting Neighnor joining #Neighnor joining is a bottom-up (agglomerative) clustering method for the creation of phylogenetic trees #Bottom-up (agglomerative) is a type of hierachical clustering where each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy

```
plot(mammals_NJ, main = "Neighbor Joining")
```



#Parsimony can be used to fit the data of the trees and compare their respective parisimony scores #optim.parismony() gives you a detailed search through the nearest-neighbor interchange (NNI), subtree pruning and regrafting (SPR). #pratchet() will perform the search with the parsimony ratchet algorithm.

```
parsimony(mammals_UPGMA, mammals10_phyDat)
```

```
## [1] 586
```

```
parsimony(mammals_NJ, mammals10_phyDat)
```

```
## [1] 580
```

```
mammals_optim <- optim.parsimony(mammals_NJ, mammals10_phyDat)
```

```
## Final p-score 580 after  0 nni operations
```

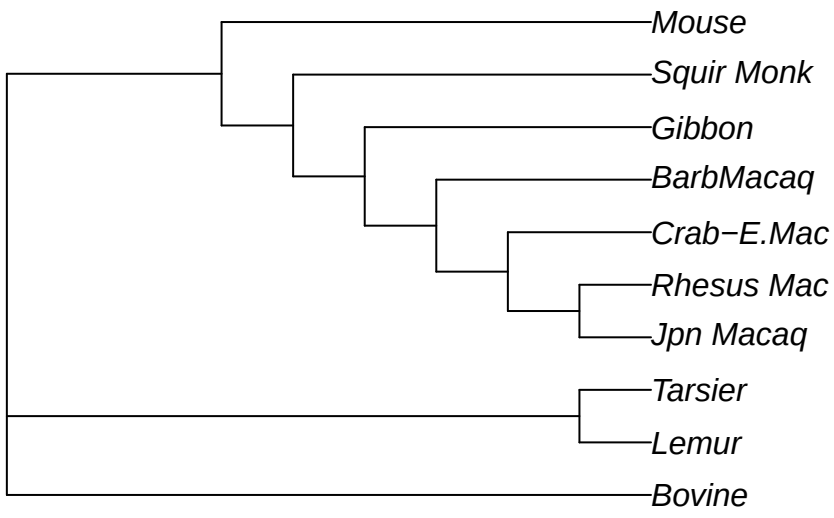
```
mammals_pratchet <- pratchet(mammals10)
```

```
## [1] "Best pscore so far: 580"
## [1] "Best pscore so far: 580"
## [1] "Best pscore so far: 580"
## [1] "Best pscore so far: 580"
## [1] "Best pscore so far: 580"
```

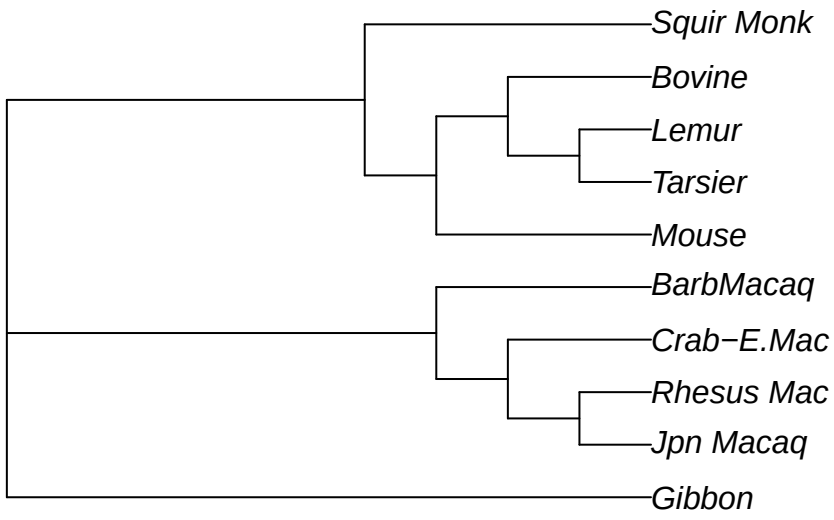
```
## [1] "Best pscore so far: 580"
## [1] "Best pscore so far: 580"
## [1] "Best pscore so far: 580"
## [1] "Best pscore so far: 580"
## [1] "Best pscore so far: 580"
## [1] "Best pscore so far: 580"
```

#plot mammals_optim and mammals_pratchet

```
plot(mammals_optim)
```




```
plot(mammals_pratchet)
```



#Maximum Likelihood and Bootstrapping #These are more computationally intensive methods than the distance matrix method #Maximum Likelihood helps you to estimate model parameters by align all your sequenced data in a statistical frame work. #pml() can beb used to compute likelihood of a given tree. #To optimize the tree topology and branch length for a selected model of nucleotide evolution, the function optim.pml() can be used

```
fit <- pml(mammals_NJ, mammals10)
print(fit)
```

```
##
##  loglikelihood: -2352.64
##
## unconstrained loglikelihood: -1230.335
##
## Rate matrix:
##   a c g t
## a 0 1 1 1
## c 1 0 1 1
## g 1 1 0 1
## t 1 1 1 0
##
## Base frequencies:
## 0.25 0.25 0.25 0.25
```

```
fitJC <- optim.pml(fit, model = "JC", rearrangement = "stochastic")
```

```
## optimize edge weights:  -2352.64 --> -2348.727
## optimize edge weights:  -2348.727 --> -2348.727
## optimize topology:  -2348.727 --> -2348.727
## 0
## [1] "Ratchet iteration  1 , best pscore so far: -2348.72667637685"
## [1] "Ratchet iteration  2 , best pscore so far: -2348.72667637685"
## [1] "Ratchet iteration  3 , best pscore so far: -2348.72667637685"
## [1] "Ratchet iteration  4 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  5 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  6 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  7 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  8 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  9 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  10 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  11 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  12 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  13 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  14 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  15 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  16 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  17 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  18 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  19 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  20 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  21 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  22 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  23 , best pscore so far: -2348.71352754093"
## [1] "Ratchet iteration  24 , best pscore so far: -2348.71352754093"
## optimize edge weights:  -2348.714 --> -2348.714
## optimize topology:  -2348.714 --> -2348.714
## 0
## optimize edge weights:  -2348.714 --> -2348.714
```

```
logLik(fitJC)
```

```
## 'log Lik.' -2348.714 (df=17)
```

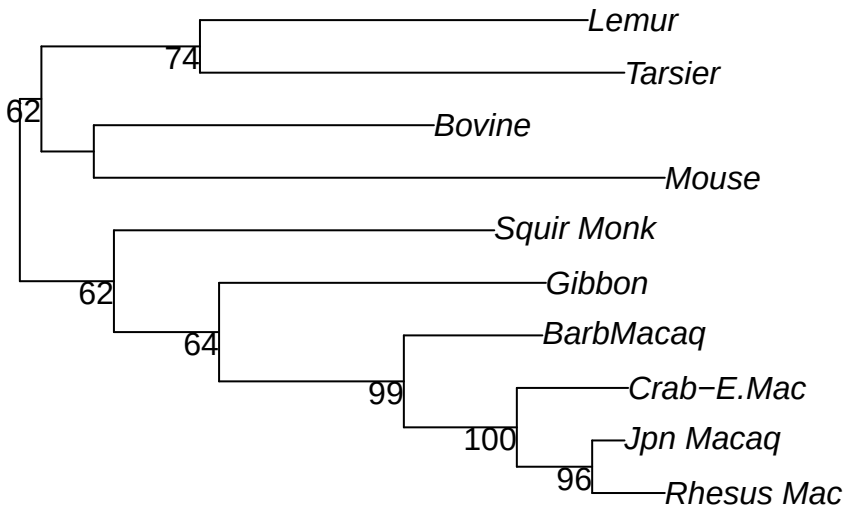
```
bs <- bootstrap.pml(fitJC, bs=100, optNni=TRUE, multicore=TRUE, control = pml.control(trace=0))
```

```
## Warning in if (!is.na(tmp)) {: the condition has length > 1 and only the first
## element will be used

## Warning in if (tmp == 1) {: the condition has length > 1 and only the first
## element will be used

## Warning in if (tmp == 2) do_rearr <- extras$rearrangement %in% c("NNI", : the
## condition has length > 1 and only the first element will be used
```

```
plotBS(midpoint(fitJC$tree), bs, p = 50, type="p")
```



#Exporting Trees #write.tree () allows you to export the output in Newick format

```
write.tree(bs, file="bootstrap_example.tre")
```