Assignment7

SiruiZHAO

2023-03-01

My GitHub (https://github.com/siruiZHAO/Assignment7-DragonPhylogenies.git)

First, we need to load library

```
library(dplyr)
library(ape)
library(Biostrings)
library(reshape2)
library(ggplot2)
library(ggtree)
```

Import nexus file to R

```
#import data
DragonData <- read.nexus.data("./input/DragonMatrix.nex")
#check data
head(DragonData)
```

```
## $`0.1FishXXX`
##  [1] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
## [20] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
## [39] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
## [58] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0"
## [77] "0" "0"
##
## $`0.2SnakeXX`
##  [1] "1" "1" "1" "1" "0" "0" "0" "0" "1" "1" "1" "0" "0" "1" "1" "1" "0" "1" "1"
## [20] "0" "0" "0" "0" "0" "0" "0" "1" "1" "0" "0" "0" "0" "0" "0" "1" "0" "0" "0"
## [39] "1" "0" "0" "0" "1" "0" "0" "0" "1" "0" "0" "0" "0" "0" "0" "1" "1" "1" "1"
## [58] "1" "1" "0" "0" "0" "0" "0" "0" "0" "0" "1" "0" "0" "0" "0" "0" "0" "0" "0"
## [77] "0" "0"
##
## $`0.3MammalX`
##  [1] "1" "0" "0" "0" "0" "0" "0" "0" "0" "0" "1" "1" "1" "0" "0" "0" "0" "0" "0"
## [20] "0" "0" "0" "1" "0" "0" "0" "1" "1" "1" "0" "0" "0" "0" "0" "1" "1" "0" "0"
## [39] "1" "1" "0" "0" "1" "1" "0" "0" "0" "0" "0" "0" "0" "0" "0" "1" "0" "0" "0"
## [58] "0" "0" "0" "1" "1" "0" "0" "0" "0" "0" "1" "0" "0" "0" "0" "0" "1" "0" "0"
## [77] "0" "0"
##
## $`1GermanXXX`
##  [1] "0" "1" "0" "0" "1" "1" "1" "0" "0" "1" "1" "1" "1" "0" "0" "0" "0" "1" "1"
## [20] "0" "0" "0" "0" "1" "0" "1" "1" "1" "1" "1" "0" "0" "0" "0" "1" "1" "0" "1"
## [39] "0" "0" "0" "0" "1" "1" "0" "1" "1" "0" "0" "0" "0" "0" "0" "1" "1" "1" "0"
## [58] "0" "0" "0" "1" "0" "0" "0" "1" "0" "0" "1" "0" "?" "?" "?" "?" "0" "0" "1"
## [77] "1" "1"
##
## $`2FrenchXXX`
##  [1] "0" "1" "0" "0" "1" "1" "0" "1" "0" "1" "1" "0" "0" "1" "1" "1" "0" "1" "0"
## [20] "0" "0" "0" "0" "0" "0" "1" "1" "1" "1" "1" "0" "0" "0" "0" "1" "1" "0" "0"
## [39] "1" "0" "0" "0" "1" "1" "0" "0" "0" "0" "0" "0" "0" "0" "0" "1" "1" "0" "0"
## [58] "0" "0" "1" "0" "0" "1" "0" "0" "0" "1" "1" "0" "0" "0" "1" "1" "0" "0" "1"
## [77] "1" "0"
##
## $`3FrenchXXX`
##  [1] "0" "1" "0" "0" "1" "1" "1" "0" "0" "0" "1" "1" "0" "0" "0" "0" "0" "0" "?"
## [20] "?" "?" "1" "0" "1" "0" "0" "1" "1" "1" "1" "0" "0" "0" "0" "1" "1" "0" "1"
## [39] "1" "1" "0" "1" "1" "1" "0" "1" "0" "0" "0" "0" "0" "0" "0" "1" "1" "1" "0"
## [58] "0" "0" "0" "1" "0" "0" "0" "1" "0" "0" "1" "0" "0" "0" "1" "1" "0" "0" "1"
## [77] "1" "1"
```

```
names(DragonData)
```

```
##  [1] "0.1FishXXX" "0.2SnakeXX" "0.3MammalX" "1GermanXXX" "2FrenchXXX"
##  [6] "3FrenchXXX" "4DutchXXXX" "5EnglishXX" "6AmericanX" "7FrenchXXX"
## [11] "8EnglishXX" "9FrenchXXX" "10FrenchXX" "11SpanishX" "12Japanese"
## [16] "13Japanese" "14Japanese" "15Japanese" "16Japanese" "17Japanese"
## [21] "18Japanese" "19Japanese" "20Japanese" "21Japanese" "22Japanese"
## [26] "23Japanese" "24Japanese" "25Japanese" "26Japanese" "27Japanese"
## [31] "28Japanese" "29Japanese" "30ItalianX" "31ItalianX" "32ItalianX"
## [36] "33XXXXXXXX" "34GermanXX" "35EnglishX" "36GermanXX" "37DutchXXX"
## [41] "38SpanishX" "39ItalianX" "40ItalianX" "41EnglishX" "42ItalianX"
## [46] "43SpanishX" "44ItalianX" "45ItalianX" "46EnglishX" "47ItalianX"
## [51] "48DutchXXX" "49IndianXX" "50Japanese" "51Japanese" "52Japanese"
## [56] "53Japanese" "54IranianX" "55IranianX" "56IranianX" "57IranianX"
## [61] "58TurkishX" "59IranianX" "60IranianX" "61TurkishX" "62TurkishX"
## [66] "63UkraineX" "64UkraineX" "65RussiaXX" "66UkraineX" "67RussiaXX"
## [71] "68GreeceXX" "69ItalianX" "70American" "71BritishX" "72BritishX"
## [76] "73BritishX" "74BritishX" "75Toothles" "76Ghidorah" "77WestDrag"
```

Distance Matrix

To do the distance matrix, we need to convert our list object to a data.frame object

```
#convert to vector -- unlist the list object
DragonDataDF <- data.frame(matrix(unlist(DragonData), ncol = 78, byrow = T))
#correct the row names
row.names(DragonDataDF) <- names(DragonData)
#check data
head(DragonDataDF)
```

```
##            X1 X2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X13 X14 X15 X16 X17 X18 X19
## 0.1FishXXX  0  0  0  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0
## 0.2SnakeXX  1  1  1  1  0  0  0  0  1   1   1   0   0   1   1   1   0   1   1
## 0.3MammalX  1  0  0  0  0  0  0  0  0   0   1   1   1   0   0   0   0   0   0
## 1GermanXXX  0  1  0  0  1  1  1  0  0   1   1   1   1   0   0   0   0   1   1
## 2FrenchXXX  0  1  0  0  1  1  0  1  0   1   1   0   0   1   1   1   0   1   0
## 3FrenchXXX  0  1  0  0  1  1  1  0  0   0   1   1   0   0   0   0   0   0   ?
##            X20 X21 X22 X23 X24 X25 X26 X27 X28 X29 X30 X31 X32 X33 X34 X35 X36
## 0.1FishXXX   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 0.2SnakeXX   0   0   0   0   0   0   0   1   1   0   0   0   0   0   0   1   0
## 0.3MammalX   0   0   0   1   0   0   0   1   1   1   0   0   0   0   0   1   1
## 1GermanXXX   0   0   0   0   1   0   1   1   1   1   1   0   0   0   0   1   1
## 2FrenchXXX   0   0   0   0   0   0   1   1   1   1   1   0   0   0   0   1   1
## 3FrenchXXX   ?   ?   1   0   1   0   0   1   1   1   1   0   0   0   0   1   1
##            X37 X38 X39 X40 X41 X42 X43 X44 X45 X46 X47 X48 X49 X50 X51 X52 X53
## 0.1FishXXX   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 0.2SnakeXX   0   0   1   0   0   0   1   0   0   0   1   0   0   0   0   0   0
## 0.3MammalX   0   0   1   1   0   0   1   1   0   0   0   0   0   0   0   0   0
## 1GermanXXX   0   1   0   0   0   0   1   1   0   1   1   0   0   0   0   0   0
## 2FrenchXXX   0   0   1   0   0   0   1   1   0   0   0   0   0   0   0   0   0
## 3FrenchXXX   0   1   1   1   0   1   1   1   0   1   0   0   0   0   0   0   0
##            X54 X55 X56 X57 X58 X59 X60 X61 X62 X63 X64 X65 X66 X67 X68 X69 X70
## 0.1FishXXX   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 0.2SnakeXX   1   1   1   1   1   1   0   0   0   0   0   0   0   0   1   0   0
## 0.3MammalX   1   0   0   0   0   0   0   1   1   0   0   0   0   0   1   0   0
## 1GermanXXX   1   1   1   0   0   0   0   1   0   0   0   1   0   0   1   0   ?
## 2FrenchXXX   1   1   0   0   0   0   1   0   0   1   0   0   0   1   1   0   0
## 3FrenchXXX   1   1   1   0   0   0   0   1   0   0   0   1   0   0   1   0   0
##            X71 X72 X73 X74 X75 X76 X77 X78
## 0.1FishXXX   0   0   0   0   0   0   0   0
## 0.2SnakeXX   0   0   0   0   0   0   0   0
## 0.3MammalX   0   0   0   1   0   0   0   0
## 1GermanXXX   ?   ?   ?   0   0   1   1   1
## 2FrenchXXX   0   1   1   0   0   1   1   0
## 3FrenchXXX   0   1   1   0   0   1   1   1
```

```
#distance matrix based on new data
DragonDis <- dist(DragonDataDF, method = 'binary')
```

```
## Warning in dist(DragonDataDF, method = "binary"): NAs introduced by coercion
```

```
DisMat <- as.matrix(DragonDis)
```

Apply trait weightings

It will have five steps to treat weightings

```
#import data
weightsDat <- read.csv("./input/Weights.csv")
```

Create a single bector of weights

```
Weights <- paste0(weightsDat$Weight, collapse = "")
Weights <- strsplit(Weights, split = "")[[1]]
```

Convert each letter to a value

we could use a custom function talked in the Tutorial with the built-in letters object:

```
LETTERS # See what LETTERS is (see also letters)
```

```
##  [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q" "R" "S"
## [20] "T" "U" "V" "W" "X" "Y" "Z"
```

```
which(LETTERS == "G")
```

```
## [1] 7
```

```
WeightsNum <- rep(NA, length(Weights))
for (i in 1:length(WeightsNum)){
  if (Weights[i] %in% LETTERS){
    WeightsNum[i] <- which(LETTERS == Weights[i]) + 9
  } else {
    WeightsNum[i] <- Weights[i]
  }
}
WeightsNum <- as.numeric(WeightsNum)
```

After this, we have a vector of weights, which should have length = 78.

```
#check length
length(WeightsNum)
```

```
## [1] 78
```

Multiply the weight value by the treat vector for each dragon

Based on the fact that our data include missing data coded as ?, so we need tomultiply the 1s, which is equivalent to replacing the 1s with their corresponding weight vaues.

```
#slice our list --- using [[]]
wtDragonData <- DragonData
for (i in 1:length(DragonData)){
  RepWeight <- DragonData[[i]] == 1
  wtDragonData[[i]][RepWeight] <- WeightsNum[RepWeight]
  RepWeight <- NA
}
```

Re-calculate our distance matrix

we need to modify our objects

```
#modify data by using same way above
wtDragonDataDF <- data.frame(matrix(unlist(wtDragonData), ncol = 78, byrow = T))
row.names(wtDragonDataDF) <- names(wtDragonData)
wtDragonDist <- dist(wtDragonDataDF, method = "euclidean")
```

```
## Warning in dist(wtDragonDataDF, method = "euclidean"): NAs introduced by
## coercion
```

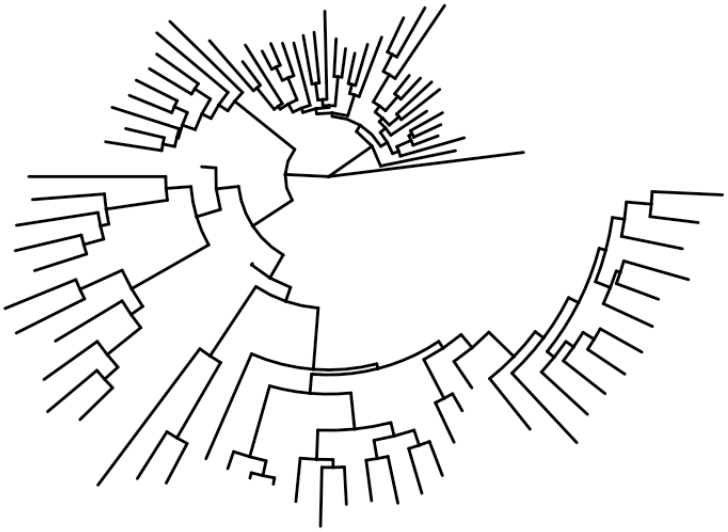
```
wtDisMat <- as.matrix(wtDragonDist)
```

plot the tree

we can compare minimal evolution (ME) and Neighbour Joining (NJ) methods

```
wtDragonTree <- fastme.bal(wtDragonDist)
wtDragonTreeNJ <- nj(wtDragonDist)
ggtree(wtDragonTree, layout = "circular")
```

```
## ! The tree contained negative edge length. If you want to ignore the edge, you
## can set `options(ignore.negative.edge=TRUE)`, then re-run ggtree.
```



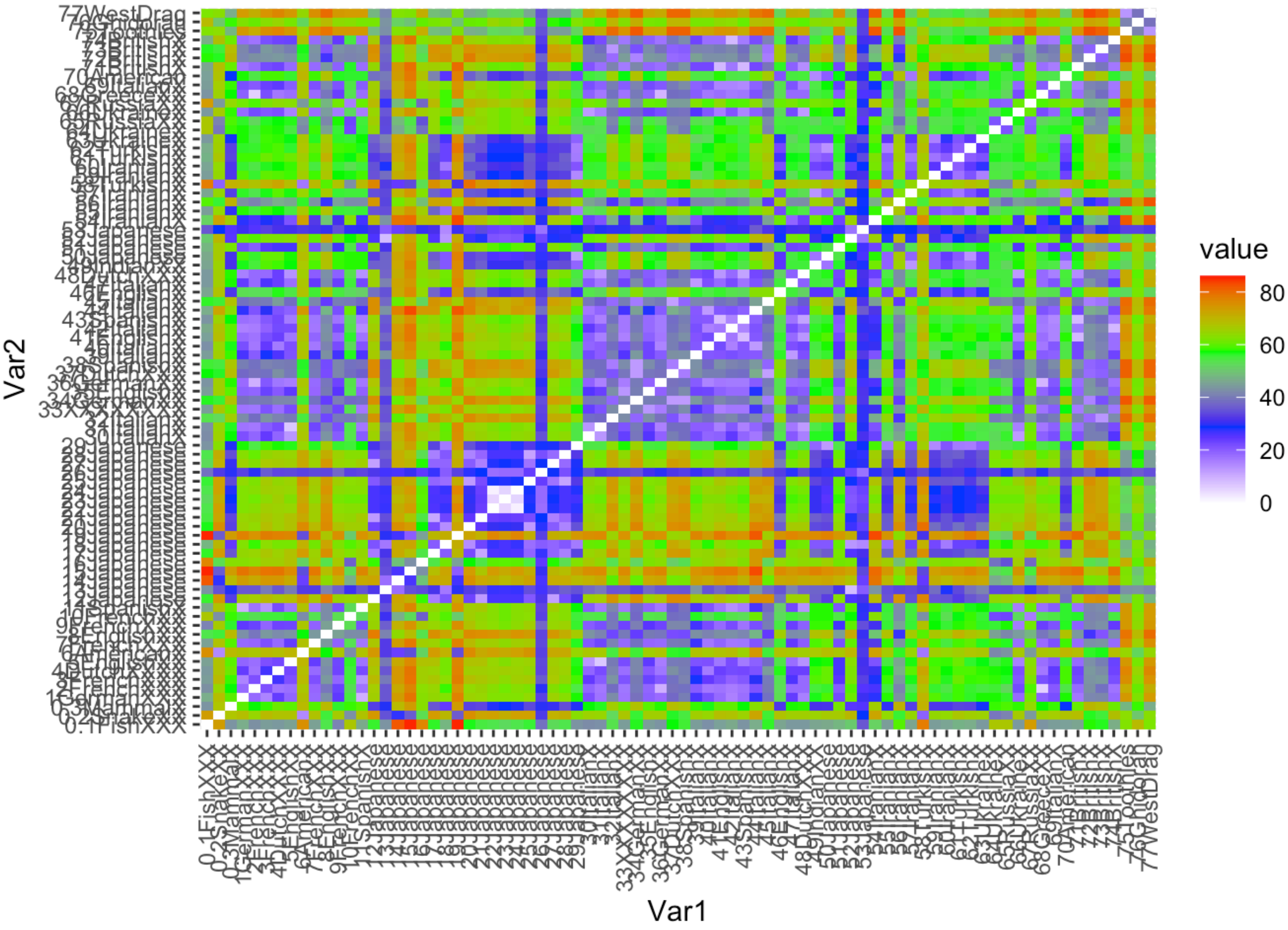
Visualize Data

To visualize the matrix in ggplot, we need rearrange the data

```
#use melt() function to rearrange data
wtPDat <- melt(wtDisMat)
```

plot the matrix

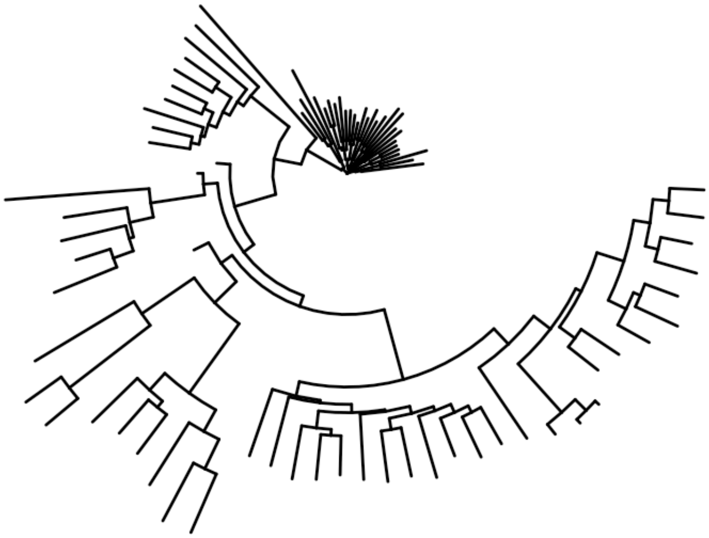
```
ggplot(data = wtPDat, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradientn(colours = c("white", "blue", "green", "red")) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5))
```



Plot the tree

We will use Neighbour Joining (NJ) methods

```
wtDragonTreeNJ <- nj(wtDragonDist)
ggtree(wtDragonTreeNJ, layout = "circular")
```



```
#check tip labels
head(wtDragonTreeNJ$tip.label)
```

```
## [1] "0.1FishXXX" "0.2SnakeXX" "0.3MammalX" "1GermanXXX" "2FrenchXXX"
## [6] "3FrenchXXX"
```

To check if dragons from same regions cluster together, we can use regular exoressions to parse out a vector:

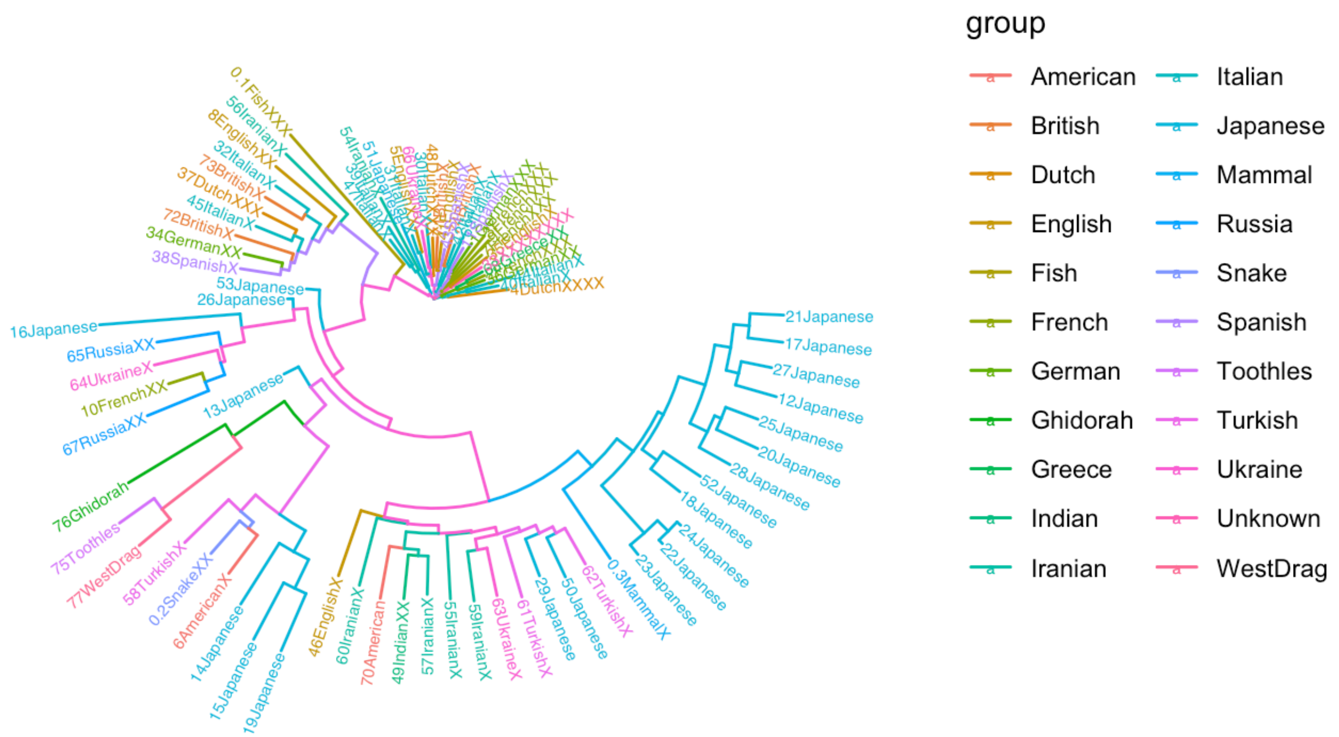
```
#remove number
Country <- gsub("[0-9\\.]+([^X]+)X*","\\1", wtDragonTreeNJ$tip.label)
Country <- gsub("\\d", "Unknown", Country)

#group tip.labels
countryGroups <- split(wtDragonTreeNJ$tip.label, Country)
names(countryGroups)
```

```
##  [1] "American" "British"  "Dutch"    "English"  "Fish"     "French"
##  [7] "German"   "Ghidorah" "Greece"   "Indian"   "Iranian"  "Italian"
## [13] "Japanese" "Mammal"   "Russia"   "Snake"    "Spanish"  "Toothles"
## [19] "Turkish"  "Ukraine"  "Unknown"  "WestDrag"
```

```
#apply group to plot
wtDTcol <- groupOTU(wtDragonTreeNJ, countryGroups)
str(wtDTcol)
```

```
## List of 4
##  $ edge       : int [1:157, 1:2] 81 89 91 104 104 91 89 100 105 105 ...
##  $ edge.length: num [1:157] 1.31 0.91 3.03 5.03 7.46 ...
##  $ tip.label  : chr [1:80] "0.1FishXXX" "0.2SnakeXX" "0.3MammalX" "1GermanXXX" ...
##  $ Nnode      : int 78
##  - attr(*, "class")= chr "phylo"
##  - attr(*, "order")= chr "cladewise"
##  - attr(*, "group")= Factor w/ 22 levels "American","British",..: 5 16 14 7 6 6 3
4 1 6 ...
```

```
#tree group by country
ggtree(wtDTcol, layout = "circular", aes(color = group)) +
  geom_tiplab(size = 2, aes(angle = angle))
```



```
#choose my dragon
wtDTcolMY <- groupOTU(wtDTcol, c(countryGroups$Toothles,
                                 countryGroups$Ghidorah,
                                 countryGroups$WestDrag))

#plot tree
ggtree(wtDTcolMY, layout = "circular", aes(color = group)) +
  geom_tiplab(size = 2, aes(angle = angle)) +
  scale_colour_manual(name = "Dragon Type",
                      values = c("blue", "red"),
                      labels = c("other", "My Dragon"))
```

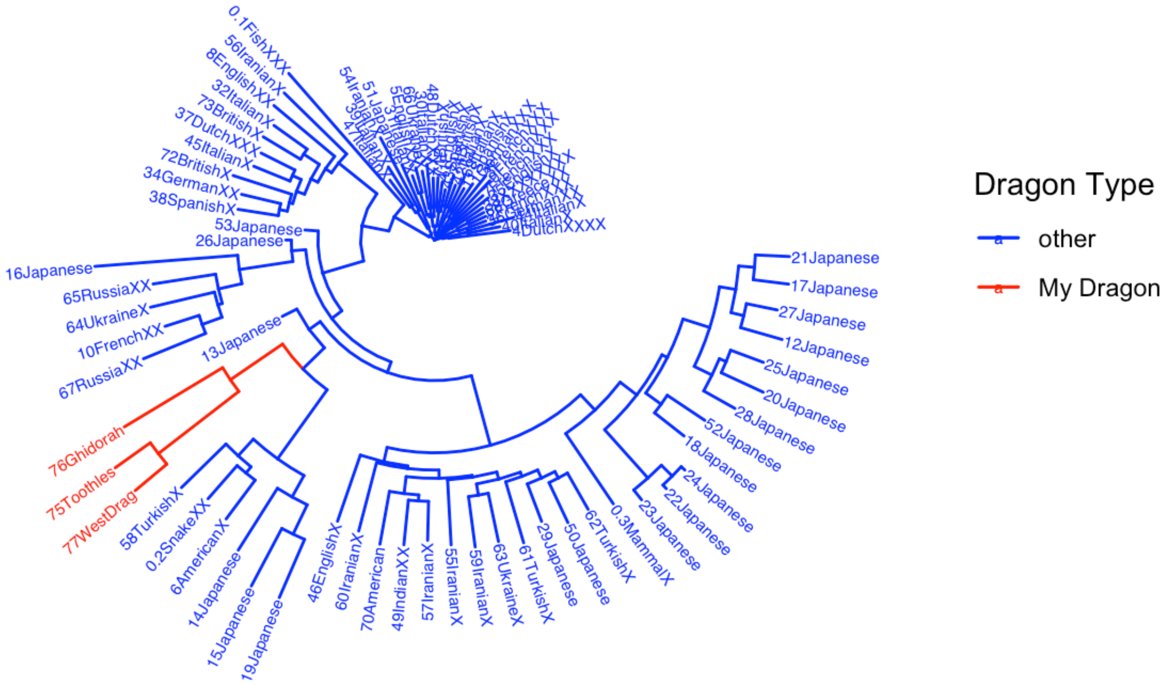


Figure 1: Phylogenetic Tree of Dragons. The three dragons in red (Toothless labeled as Toothles, King Ghidorah labeled as Ghidorah, and typical west dragon labeled as WestDrag) are dragons I choose.

Report

Toothless (Figure 2) is the only Night Fury seen thus far. He is the most intelligent of all known dragons, and seems to have a dry sense of humor, similar to that of Hiccup. Except for no tooth, his tail is like Split Y.

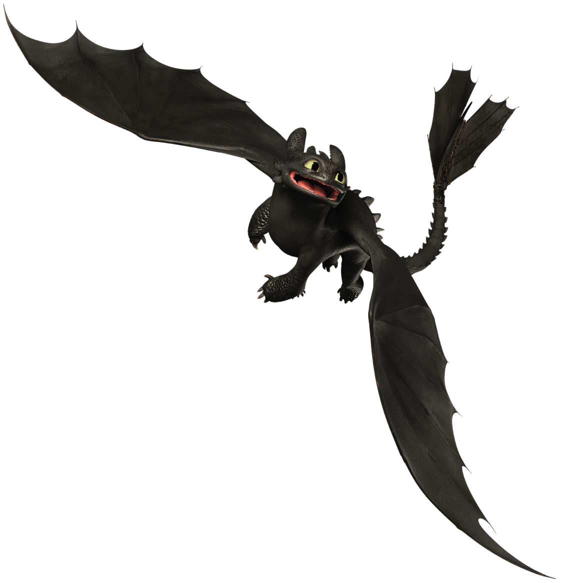


Figure 2: Toothless. Credit to Thorndrum: https://howtotrainyourdragon.fandom.com/wiki/Toothless_(Franchise)?file=Dragon+01+toothless.png (https://howtotrainyourdragon.fandom.com/wiki/Toothless_(Franchise)?file=Dragon+01+toothless.png)

King Ghidorag in Figure 3, also known as Monster Zero, is a three headed dragon kaiju appeared in film Ghidorah, the Three-Headed Monster. He arrived on Earth inside a meteor and planed to destroy it just like he had done to Mars thousands of years in the past.

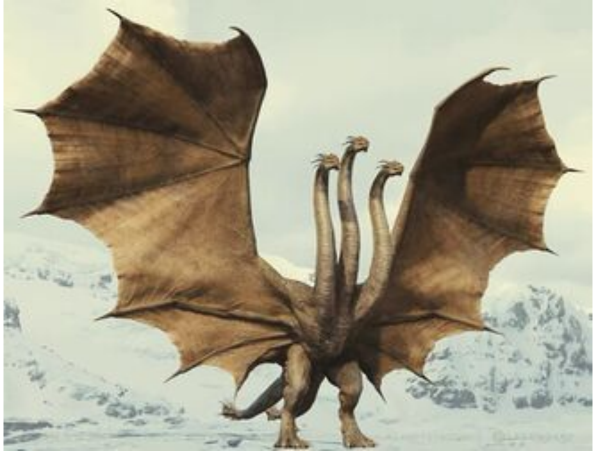


Figure 3: King Ghidorag. Credit to Astounding Beyond Belief: https://wikizilla.org/wiki/File:Vlcsnap-2021-10-22-09h36m21s456.png.jpg (https://wikizilla.org/wiki/File:Vlcsnap-2021-10-22-09h36m21s456.png.jpg)

Finally, Figure 4 is one kind of typical west dragon. Compare to King Ghidorag, Toothless is more like a west dragon and they share great similarity

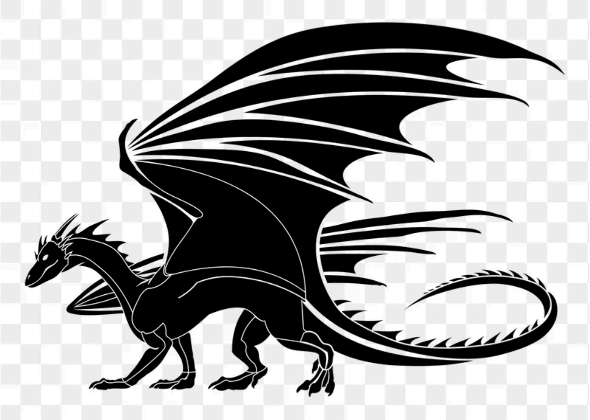


Figure 4: Typical West Dragon. Credit to rawpixel: https://www.rawpixel.com/image/6482373/png-sticker-public-domain (https://www.rawpixel.com/image/6482373/png-sticker-public-domain)

However, when considering the evolving rate of different traits and based on Figure 1, we can find that all these three dragons are closely related. But Toothless and Typical West Dragon are much closer and have a larger probability they may share a common ancestor. The largest difference between the three dragons is that King Ghidorah has three heads and two tails even though they share many similarities in other traits.

Thanks!