evidence2

Alejandro Benítez Bravo A01712835

2025-05-02

Load all of the libraries that we'll be using for this project

```
##
##    /// adegenet 2.1.11 is loaded /////////////
##
##    > overview: '?adegenet'
##    > tutorials/doc/questions: 'adegenetWeb()'
##    > bug reports/feature requests: adegenetIssues()
```

```
## Loading required package: BiocGenerics
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## 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
```

```
## Loading required package: stats4
```

```
##
## Attaching package: 'S4Vectors'
```

```
## 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
```

```
## Loading required package: XVector
```

```
## Loading required package: GenomeInfoDb
##
## Attaching package: 'Biostrings'
## The following object is masked from 'package:ape':
##
##     complement
## The following object is masked from 'package:base':
##
##     strsplit
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:Biostrings':
##
##     collapse, intersect, setdiff, setequal, union
## The following object is masked from 'package:GenomeInfoDb':
##
##     intersect
## The following object is masked from 'package:XVector':
##
##     slice
## The following objects are masked from 'package:IRanges':
##
##     collapse, desc, intersect, setdiff, slice, union
## The following objects are masked from 'package:S4Vectors':
##
##     first, intersect, rename, setdiff, setequal, union
## The following objects are masked from 'package:BiocGenerics':
##
##     combine, intersect, setdiff, union
## 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
##
## Attaching package: 'tidyr'
## The following object is masked from 'package:reshape2':
##
##     smiths
## The following object is masked from 'package:S4Vectors':
##
##     expand
```

```
##
## Attaching package: 'phangorn'

## The following object is masked from 'package:adegenet':
##
##     AICc

## ggtree v3.14.0 Learn more at https://yulab-smu.top/contribution-tree-data/
##
## Please cite:
##
## S Xu, Z Dai, P Guo, X Fu, S Liu, L Zhou, W Tang, T Feng, M Chen, L
## Zhan, T Wu, E Hu, Y Jiang, X Bo, G Yu. ggtreeExtra: Compact
## visualization of richly annotated phylogenetic data. Molecular Biology
## and Evolution. 2021, 38(9):4039-4042. doi: 10.1093/molbev/msab166

##
## Attaching package: 'ggtree'

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

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

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

## The following object is masked from 'package:S4Vectors':
##
##     expand

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

## treeio v1.30.0 Learn more at https://yulab-smu.top/contribution-tree-data/
##
## Please cite:
##
## LG Wang, TTY Lam, S Xu, Z Dai, L Zhou, T Feng, P Guo, CW Dunn, BR
## Jones, T Bradley, H Zhu, Y Guan, Y Jiang, G Yu. treeio: an R package
## for phylogenetic tree input and output with richly annotated and
## associated data. Molecular Biology and Evolution. 2020, 37(2):599-603.
## doi: 10.1093/molbev/msz240

##
## Attaching package: 'treeio'

## The following object is masked from 'package:Biostrings':
##
##     mask
```

Performing sequence loading

```
# List of countries where COVID killed the most people
countries <- c("usa", "china", "india", "france", "germany", "brazil", "south_korea", "japan", "italy",
```

```r
# Map the names to DNA String Sets
sequences <- sapply(countries, function(x) readDNAStringSet(paste("./sequences/", x, ".fasta", sep="")))

# Convert to DNABin types
dna_bins <- do.call(c, sapply(sequences, as.DNAbin))

# Labeling and Display
country_titles <- sapply(countries, function(string) str_to_title(str_replace(string, "_", " ")))

# Create a  Dataframe from the Sequences
sequence_frame <- data.frame(
  country_key = countries,
  country_title  = country_titles,
  sequence_length = sapply(sequences, width),
  a_frequency = sapply(sequences, function(seq) letterFrequency(seq, 'A')),
  c_frequency = sapply(sequences, function(seq) letterFrequency(seq, 'C')),
  g_frequency = sapply(sequences, function(seq) letterFrequency(seq, 'G')),
  t_frequency = sapply(sequences, function(seq) letterFrequency(seq, 'T'))
)

# Print Genome Sequence Lengths
print("The sequence lengths for each genome are as follows")
```

```
## [1] "The sequence lengths for each genome are as follows"
```

```r
sequence_frame[3]
```

```
##             sequence_length
## usa                   29750
## china                 29903
## india                 29834
## france                29759
## germany               29741
## brazil                29835
## south_korea           29738
## japan                 29784
## italy                 29741
## uk                    29750
```

```r
# Print Genome Sequence Nitrogenous Base Frequencies
print("The frequencies for each base in each genome are as follows:")
```

```
## [1] "The frequencies for each base in each genome are as follows:"
```

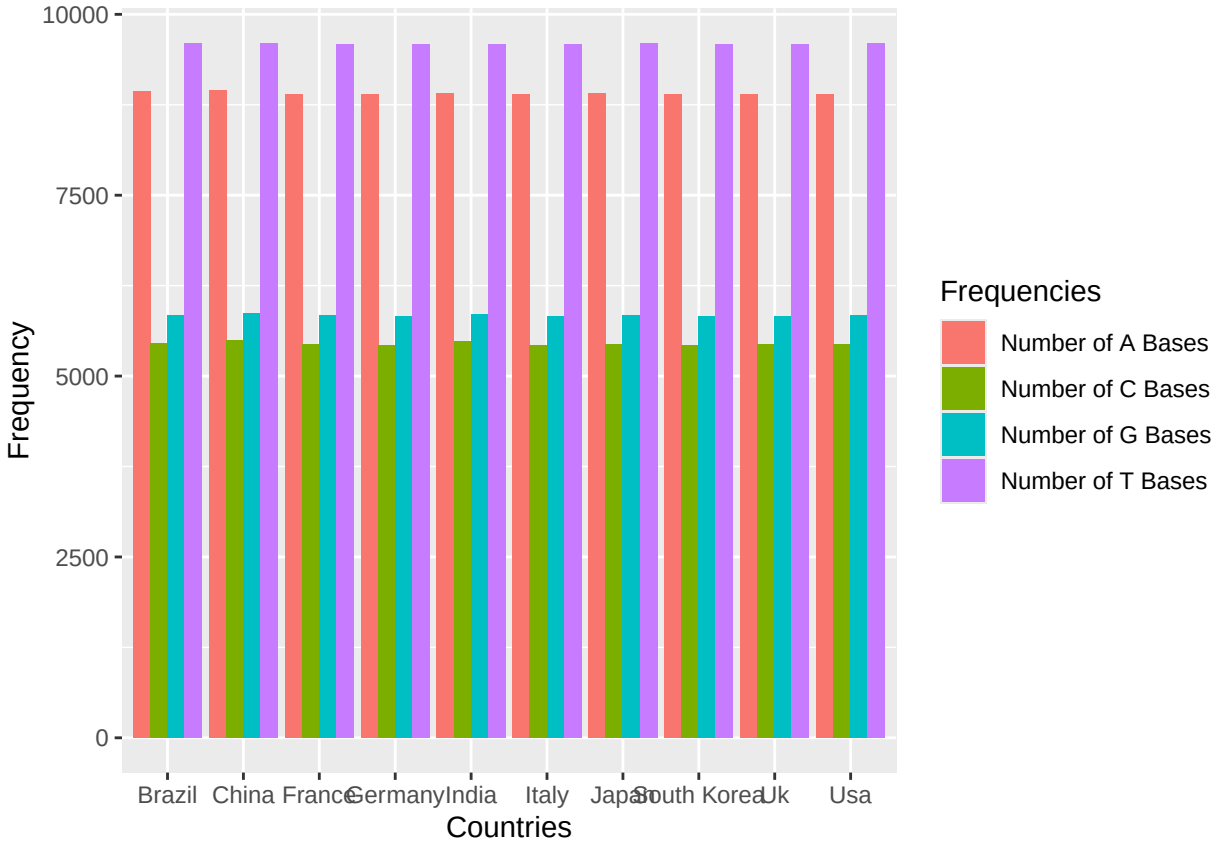
```r
sequence_frame[c(4, 5, 6, 7)]
```

```
##             a_frequency c_frequency g_frequency t_frequency
## usa                8890        5433        5834        9593
## china              8954        5492        5863        9594
## india              8909        5482        5851        9592
## france             8895        5443        5834        9587
## germany            8896        5429        5828        9588
## brazil             8943        5457        5841        9594
## south_korea        8891        5429        5829        9589
## japan              8905        5441        5834        9604
```

```
## italy               8893        5425        5830        9592
## uk                  8893        5433        5830        9592
```

Convert to data frame and graph

```
# Plot Frequencies
ggplot(
  data = sequence_frame %>% gather(Frequencies, Frequency, -country_title, -country_key, -sequence_leng
  aes(x = country_title, y = Frequency, fill = Frequencies)
) +
  geom_bar(stat = 'identity', position = 'dodge') +
  xlab("Countries") +
  scale_fill_discrete(
    labels = c(
      "a_frequency" = "Number of A Bases",
      "c_frequency" = "Number of C Bases",
      "g_frequency" = "Number of G Bases",
      "t_frequency" = "Number of T Bases"
  ))
```



En esta gráfica se pueden observar la comparación entre todas las bases de cada variante por país. Realizando un análisis simplemente de esta gráfica nos podemos dar cuenta de que las variantes son muy parecidas. Si bien esta gráfica no nos permite ver exáctamente cuáles son las diferencias, se puede apreciar que las variantes son mayormente similares, lo cual puede revelar el hecho de que fue una variante la que se propagó por todo el mundo, y fue en cada lugar donde fue mutando pero desde la misma base.

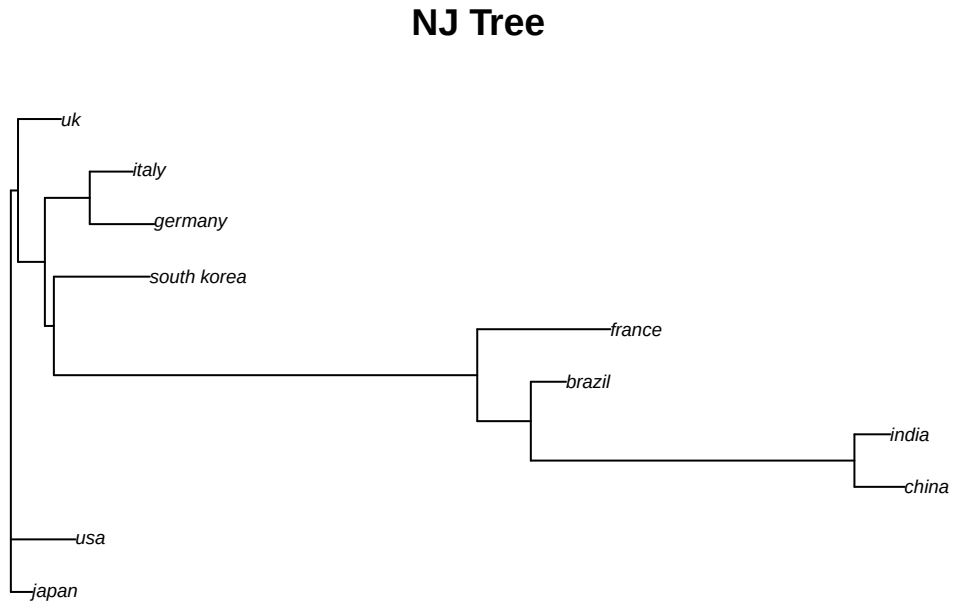
```
# Read FASTAs Again but in a Different Format (idealy, this shouldn't be the case but we couldn't find
countries <- c("usa", "china", "india", "france", "germany", "brazil", "south_korea", "japan", "italy",
```

```r
# Place Sequences into a DNAStringSet
all_sequences <- DNAStringSet()
for (country in countries) {
  seq <- readDNAStringSet(paste0("./sequences/", country, ".fasta"))
  names(seq) <- country  # Explicitly name each sequence
  all_sequences <- c(all_sequences, seq)
}

# Load Mortality Rates
# 3. Prepare mortality data with EXACT MATCH to tip labels
mortality_data <- data.frame(
  label = countries,  # Must match tree$tip.label
  Cases = c(103436829, 99380363, 45043415, 39016278, 38437756, 37511921,
            34571873, 33803572, 26826486, 24992089),
  Deaths = c(1201488, 122358, 533641, 168091, 174979, 702116,
             35934, 74694, 197542, 232112)

) %>%
  mutate(
    Mortality = round(Deaths/Cases * 100, 2),
    Country = stringr::str_to_title(gsub("_", " ", label))
  )

mortality_data$Rates = mortality_data$Deaths / mortality_data$Cases

# Align DNA Sequences
aligned <- AlignSeqs(all_sequences)
```

```
## Determining distance matrix based on shared 11-mers:
## ============================================================================
##
## Time difference of 0.02 secs
##
## Clustering into groups by similarity:
## ============================================================================
##
## Time difference of 0 secs
##
## Aligning Sequences:
## ============================================================================
##
## Time difference of 0.2 secs
##
## Iteration 1 of 2:
##
## Determining distance matrix based on alignment:
## ============================================================================
##
## Time difference of 0 secs
##
## Reclustering into groups by similarity:
## ============================================================================
##
## Time difference of 0 secs
```

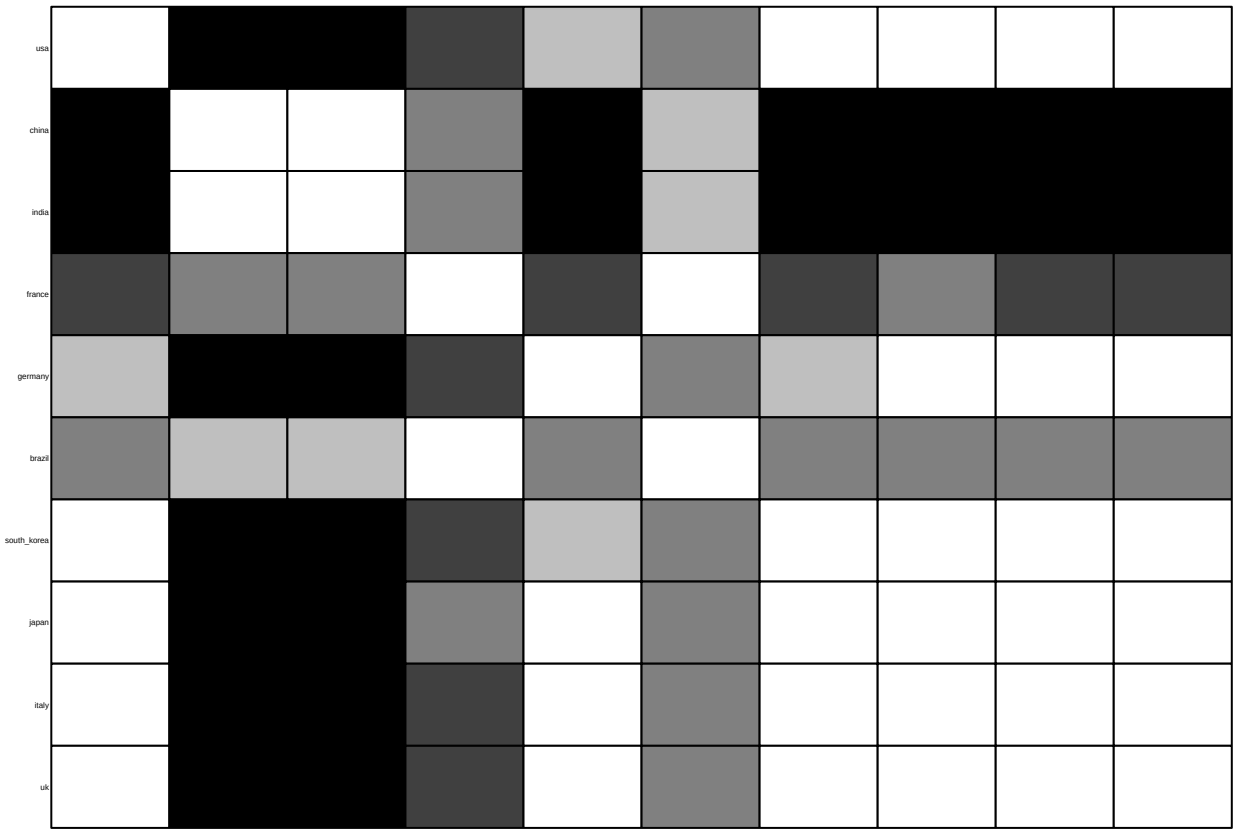
```
##
## Realigning Sequences:
## ============================================================================
##
## Time difference of 0 secs
##
## Alignment converged - skipping remaining iteration.
```

```r
dna <- as.matrix(as.DNAbin(aligned))

# Stop if Our Sequences are not Aligned
stopifnot(length(unique(ncol(dna))) == 1)

# DNA Distribution
dna_distro <- dist.dna(dna, model = "TN93")

# Create NJ Tree From DNA Distro
phylotree <- nj(dna_distro)



# Calculate Bootstrapping Values
boots <- boot.phylo(
  phylotree,
  dna,
  function(e) root(nj(dist.dna(e, model = "TN93")), 1),
  B = 100,
  quiet = TRUE
)

# Plot our Phylo Tree with Bootstrap Values
myPal <- colorRampPalette(c("red","yellow","green","blue"))
mortalityPalette <- colorRampPalette(c("red", "blue"))
plot(phylotree, cex = 0.6, main = "NJ Tree")
```



Este es nuestro árbol filogenético simple. Este árbol está hecho por medio de `ape`. Es interesante observar este árbol, ya que nos revela que las variantes más lejanas y las que más mutaron fueron las de China e India. Esto puede ser debido a el acomodo y orden en el que se crea el árbol filogenético, pero también podría ser porque las variantes que obtuvimos para China e India fueron cuándo el virus ya había mutado bastante.

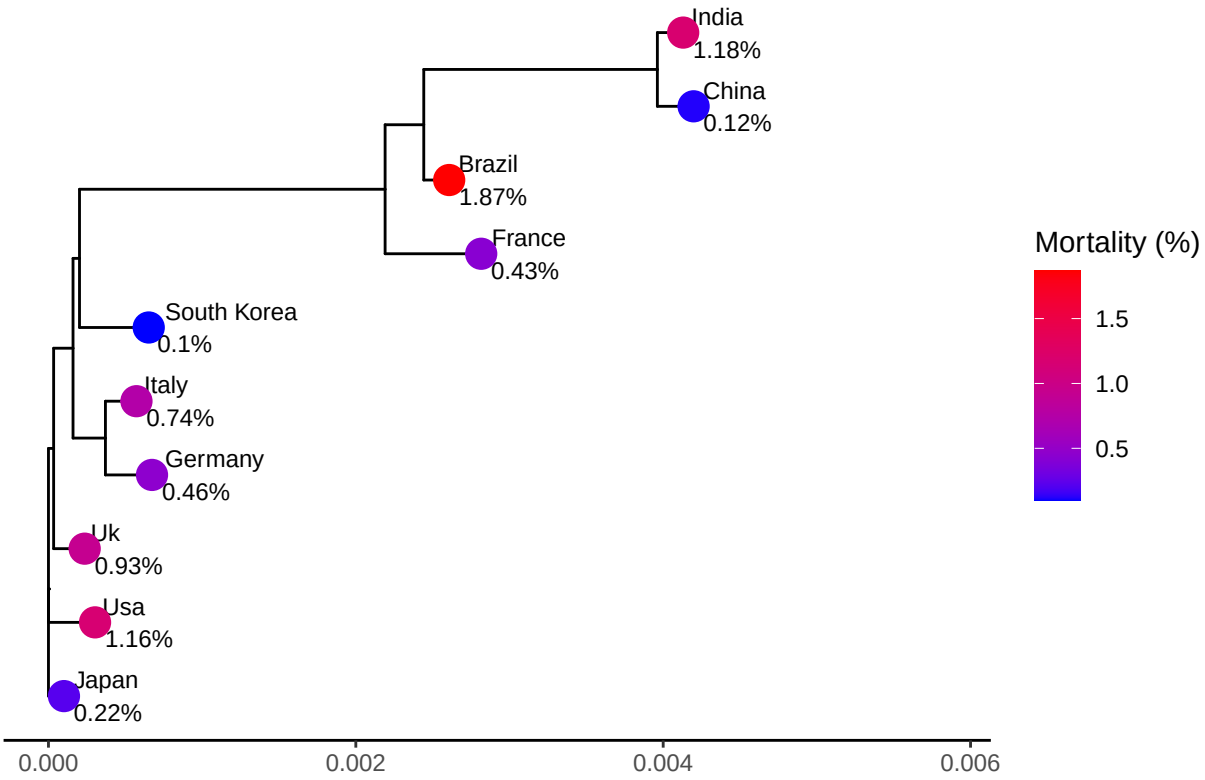
```
# Convert our DNA Distribution and Plot it in a Table
adj_matrix <- as.matrix(dna_distro)
table.paint(
  adj_matrix,
  cleg = 0,
  clabel.row = 0.25,
  clabel.col = 0.25
)
```



Aquí podemos ver la matriz de adyacencia. Naturalmente la diagonal es toda blanca ya que se comparan las mismas variantes. Podemos observar que las variantes de China e India son muy parecidas entre sí. Las variantes de Korea del Sur, Japón, Italia, y el Reino unido también son bastante similares. A nivel de continentes, no parece haber un patrón. Japon y Korea del Sur tiene variantes muy diferentes a China e India, países que se encuentran en Asia, pero como lo vimos anteriormente muy similares a países de Europa.

```
p <- ggtree(phylotree) %<+% mortality_data +
  geom_tiplab(aes(label = paste0(Country, "\n", Mortality, "%")),
              size = 3, hjust = -0.1) +
  geom_tippoint(aes(color = Mortality), size = 5) +
  scale_color_gradient(low = "blue", high = "red", na.value = "gray") +
  xlim(0, max(dna_distro) * 1.3) +
  labs(title = "Phylogenetic Tree with Mortality Rates",
       color = "Mortality (%)") +
  theme_tree2()
```

```
print(p)
```



Para mi aportación extra decidí hacer dos cosas: analizar la mortalidad de las variantes para ver si las mutaciones afectaban esto, y usar ggtree para generar el árbol filogenético (parcialmente porque batallé mucho con ape y me resultó más fácil usar ggtree).

Para la mortalidad no se hizo un análisis tan profundo. Simplemente se buscaron la cantidad de casos y las muertes en cada país, y se asumió que esos eran los datos para las variantes de cada país. Con ggtree simplemente se hizo un árbol con colores para demostrar el nivel de mortalidad, además de que se añadió el porcentaje de mortalidad.

Esto nos arroja resultados muy interesantes. Se puede observar que las variantes con menos mutaciones tiene porcentajes ligeramente más bajos, o colores más fríos. Sin embargo esto no es del todo cierto, ya que Estados Unidos muestra una mortalidad considerable. Sin embargo los demás parecer ser medianamente bajos. Y podemos observar que los países con más mutaciones, como Brazil e India, tienen una alta mortalidad. Sin embargo, también se puede observar que Francia y China tiene muy baja mortalidad comparandolo con los demás.