DNA Alignments Assignment

Roshael Chellappah (20103016)

02/03/2022

GitHub Username: RoshaelC Repository Link: https://github.com/RoshaelC/DNAalignment.git (https://github.com/RoshaelC/DNAalignment.git)

Purpose of Analysis: Odd sequence of non-human DNA found from nanopore sequencing of patient biofluids. Aim to identify the origins of this sequence and potential impact (or lack thereof) of identified sequence on patient.

Load required packages:

```
library(BiocManager)
library(sangerseqR)
```

```
## Loading required package: Biostrings
```

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

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

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, 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, setdiff, sort, table, tapply,
##     union, unique, unsplit, which.max, which.min
```

```
## Loading required package: S4Vectors
```

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

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

```
## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname
```

```
## Loading required package: IRanges
```

```
##
## Attaching package: 'IRanges'
```

```
## The following object is masked from 'package:grDevices':
##
##     windows
```

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

```
## Loading required package: GenomeInfoDb
```

```
##
## Attaching package: 'Biostrings'
```

```
## The following object is masked from 'package:base':
##
##     strsplit
```

```
library(rentrez)
library(genbankr)
```

Sequence Analysis

Sequence preparation for analysis:

```
UnknownID <- c("ATGTCTGATAATGGACCCCAAAATCAGCGAAATGCACCCCGCATTACGTTTGGTGGACCCTCAGATTCAACTGGCAGTAA
CCAGAATGGAGAACGCAGTGGGGCGCGATCAAAACAACGTCGGCCCCAAGGTTTACCCAATAATACTGCGTCTTGGTTCACCGCTCTCACTCAACA
TGGCAAGGAAGACCTTAAATTCCCTCGAGGACAAGGCGTTCCAATTAACACCAATAGCAGTCCAGATGACCAAATTGGCTACTACCGAAGAGCTAC
CAGACGAATTCGTGGTGGTGACGGTAAAATGAAAGATCTCAGTCCAAGATGGTATTTCTACTACCTAGGAACTGGGCCAGAAGCTGGACTTCCCTA
TGGTGCTAACAAAGACGGCATCATATGGGTTGCAACTGAGGGAGCCTTGAATACACCAAAAGATCACATTGGCACCCGCAATCCTGCTAACAATGC
TGCAATCGTGCTACAACTTCCTCAAGGAACAACATTGCCAAAAGGCTTCTACGCAGAAGGGAGCAGAGGCGGCAGTCAAGCCTCTTCTCGTTCCTC
ATCACGTAGTCGCAACAGTTCAAGAAATTCAACTCCAGGCAGCAGTAGGGGAACTTCTCCTGCTAGAATGGCTGGCAATGGCGGTGATGCTGCTCT
TGCTTTGCTGCTGCTTGACAGATTGAACCAGCTTGAGAGCAAAATGTCTGGTAAAGGCCAACAACAACAAGGCCAAACTGTCACTAAGAAATCTGC
TGCTGAGGCTTCTAAGAAGCCTCGGCAAAAACGTACTGCCACTAAAGCATACAATGTAACACAAGCTTTCGGCAGACGTGGTCCAGAACAAACCCA
AGGAAATTTTGGGGACCAGGAACTAATCAGACAAGGAACTGATTACAAACATTGGCCGCAAATTGCACAATTTGCCCCCAGCGCTTCAGCGTTCTT
CGGAATGTCGCGCATTGGCATGGAAGTCACACCTTCGGGAACGTGGTTGACCTACACAGGTGCCATCAAATTGGATGACAAAGATCCAAATTTCAA
AGATCAAGTCATTTTGCTGAATAAGCATATTGACGCATACAAAACATTCCCACCAACAGAGCCTAAAAAGGACAAAAAGAAGAAGGCTGATGAAAC
TCAAGCCTTACCGCAGAGACAGAAGAAACAGCAAACTGTGACTCTTCTTCCTGCTGCAGATTTGGATGATTTCTCCAAACAATTGCAACAATCCAT
GAGCAGTGCTGACTCAACTCAGGCCTAA") # turn sequence into an object

class(UnknownID) # check that it is a character
```

```
## [1] "character"
```

Run BLAST search:

```
library(annotate)
```

```
## Loading required package: AnnotationDbi
```

```
## Loading required package: Biobase
```

```
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
## Loading required package: XML
```

```
UnknownBLAST <- blastSequences(UnknownID, as = 'data.frame', hitListSize = 20, timeout = 600) #
 find related sequences in NCBI's database
```

```
## estimated response time 103 seconds
```

```
## elapsed time 104 seconds
```

Multiple Alignments

```
library(ape)
```

```
##
## Attaching package: 'ape'
```

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

```
# make a vector of accession numbers from the BLAST results above and make them into a data.fram
e object

UnknownHitsDF <- data.frame(ID = UnknownBLAST$Hit_accession, Seq = UnknownBLAST$Hsp_hseq, string
sAsFactors = FALSE)
```

```
# read a sample of the sequences from GenBank
UnknownHitSeq <- read.GenBank(UnknownBLAST$Hit_accession)

# check the species
attr(UnknownHitSeq, "species")
```

```
##  [1] "Severe_acute_respiratory_syndrome_coronavirus_2"
##  [2] "Severe_acute_respiratory_syndrome_coronavirus_2"
##  [3] "Severe_acute_respiratory_syndrome_coronavirus_2"
##  [4] "Severe_acute_respiratory_syndrome_coronavirus_2"
##  [5] "Severe_acute_respiratory_syndrome_coronavirus_2"
##  [6] "Severe_acute_respiratory_syndrome_coronavirus_2"
##  [7] "Severe_acute_respiratory_syndrome_coronavirus_2"
##  [8] "Severe_acute_respiratory_syndrome_coronavirus_2"
##  [9] "Severe_acute_respiratory_syndrome_coronavirus_2"
## [10] "Severe_acute_respiratory_syndrome_coronavirus_2"
## [11] "Severe_acute_respiratory_syndrome_coronavirus_2"
## [12] "Severe_acute_respiratory_syndrome_coronavirus_2"
## [13] "Severe_acute_respiratory_syndrome_coronavirus_2"
## [14] "Severe_acute_respiratory_syndrome_coronavirus_2"
## [15] "Severe_acute_respiratory_syndrome_coronavirus_2"
## [16] "Severe_acute_respiratory_syndrome_coronavirus_2"
## [17] "Severe_acute_respiratory_syndrome_coronavirus_2"
## [18] "Severe_acute_respiratory_syndrome_coronavirus_2"
## [19] "Severe_acute_respiratory_syndrome_coronavirus_2"
## [20] "Severe_acute_respiratory_syndrome_coronavirus_2"
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:AnnotationDbi':
##
##     select
```

```
## The following object is masked from 'package:Biobase':
##
##     combine
```

```
## 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 objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(Biostrings)

UnknownDNAstring <- UnknownHitsDF$Seq %>% as.character %>% lapply(., paste0, collapse = "") %>%
 unlist %>% DNAStringSet()
```

```
# convert to a new object and add index number
names(UnknownDNAstring)<-paste(1:nrow(UnknownHitsDF),UnknownHitsDF$ID,sep="_")

# run muscle() on DNAStringSet object
library(muscle)
```

```
##
## Attaching package: 'muscle'
```

```
## The following object is masked from 'package:ape':
##
##     muscle
```

```
UnknownAlign <- muscle::muscle(stringset = UnknownDNAstring, quite = T)
```

```
## Invalid option "quite"
##
## MUSCLE v3.8.31 by Robert C. Edgar
##
## http://www.drive5.com/muscle
## This software is donated to the public domain.
## Please cite: Edgar, R.C. Nucleic Acids Res 32(5), 1792-97.
##
##
## Basic usage
##
##     muscle -in <inputfile> -out <outputfile>
##
## Common options (for a complete list please see the User Guide):
##
##     -in <inputfile>    Input file in FASTA format (default stdin)
##     -out <outputfile>  Output alignment in FASTA format (default stdout)
##     -diags             Find diagonals (faster for similar sequences)
##     -maxiters <n>      Maximum number of iterations (integer, default 16)
##     -maxhours <h>      Maximum time to iterate in hours (default no limit)
##     -html              Write output in HTML format (default FASTA)
##     -msf               Write output in GCG MSF format (default FASTA)
##     -clw               Write output in CLUSTALW format (default FASTA)
##     -clwstrict         As -clw, with 'CLUSTAL W (1.81)' header
##     -log[a] <logfile>  Log to file (append if -loga, overwrite if -log)
##     -quiet             Do not write progress messages to the screen
##     -version           Display version information and exit
##
## Without refinement (very fast, avg accuracy similar to T-Coffee): -maxiters 2
## Fastest possible (amino acids): -maxiters 1 -diags -sv -distance1 kbit20_3
## Fastest possible (nucleotides): -maxiters 1 -diags
## file504c1f107f53 20 seqs, max length 1260, avg  length 1260
## 1 MB(0%)00:00:00                Iter   1    0.48%  K-mer dist pass 1
72 MB(3%)00:00:00               Iter   1  100.00%  K-mer dist pass 1
## 72 MB(3%)00:00:00               Iter   1    0.48%  K-mer dist pass 2
72 MB(3%)00:00:00               Iter   1  100.00%  K-mer dist pass 2
## 1279 MB(52%)00:00:00               Iter   1    5.26%  Align node
1279 MB(52%)00:00:00               Iter   1   10.53%  Align node
1279 MB(52%)00:00:00               Iter   1   15.79%  Align node
1279 MB(52%)00:00:00               Iter   1   21.05%  Align node
1279 MB(52%)00:00:00               Iter   1   26.32%  Align node
1279 MB(52%)00:00:00               Iter   1   31.58%  Align node
1279 MB(52%)00:00:00               Iter   1   36.84%  Align node
1279 MB(52%)00:00:00               Iter   1   42.11%  Align node
1279 MB(52%)00:00:00               Iter   1   47.37%  Align node
1279 MB(52%)00:00:00               Iter   1   52.63%  Align node
1279 MB(52%)00:00:00               Iter   1   57.89%  Align node
1279 MB(52%)00:00:00               Iter   1   63.16%  Align node
1279 MB(52%)00:00:00               Iter   1   68.42%  Align node
1279 MB(52%)00:00:00               Iter   1   73.68%  Align node
1279 MB(52%)00:00:00               Iter   1   78.95%  Align node
1279 MB(52%)00:00:00               Iter   1   84.21%  Align node
```

```
1279 MB(52%)00:00:00               Iter   1   89.47%  Align node
1279 MB(52%)00:00:00               Iter   1   94.74%  Align node
1279 MB(52%)00:00:00               Iter   1  100.00%  Align node
1279 MB(52%)00:00:00               Iter   1  100.00%  Align node
## 1279 MB(52%)00:00:00                 Iter   1    5.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   10.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   15.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   20.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   25.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   30.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   35.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   40.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   45.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   50.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   55.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   60.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   65.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   70.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   75.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   80.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   85.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   90.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1   95.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1  100.00%  Root alignment
1279 MB(52%)00:00:00               Iter   1  100.00%  Root alignment
## 1279 MB(52%)00:00:00                 Iter   2  100.00%  Root alignment
## 1279 MB(52%)00:00:00                 Iter   3    5.41%  Refine biparts
1279 MB(52%)00:00:00               Iter   3    8.11%  Refine biparts
1279 MB(52%)00:00:00               Iter   3   10.81%  Refine biparts
1279 MB(52%)00:00:00               Iter   3   13.51%  Refine biparts
1279 MB(52%)00:00:00               Iter   3   16.22%  Refine biparts
1279 MB(52%)00:00:00               Iter   3   18.92%  Refine biparts
1279 MB(52%)00:00:00               Iter   3   21.62%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   24.32%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   27.03%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   29.73%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   32.43%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   35.14%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   37.84%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   40.54%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   43.24%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   45.95%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   48.65%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   51.35%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   54.05%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   56.76%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   59.46%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   62.16%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   64.86%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   67.57%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   70.27%  Refine biparts
1279 MB(52%)00:00:01               Iter   3   72.97%  Refine biparts
```

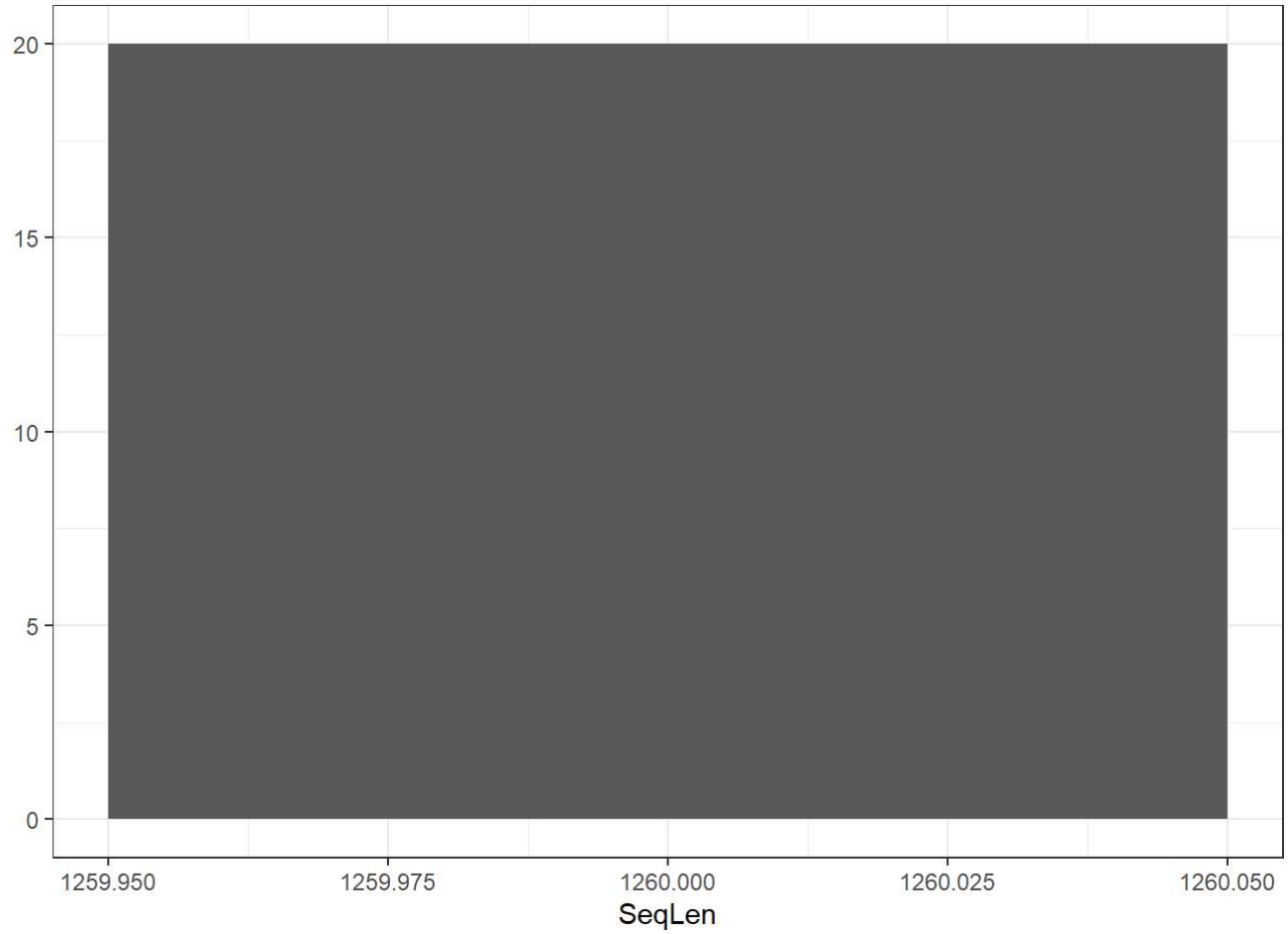
```
1279 MB(52%)00:00:01                Iter   3   75.68%  Refine biparts
1279 MB(52%)00:00:01                Iter   3   78.38%  Refine biparts
1279 MB(52%)00:00:01                Iter   3   81.08%  Refine biparts
1279 MB(52%)00:00:01                Iter   3   83.78%  Refine biparts
1279 MB(52%)00:00:01                Iter   3   86.49%  Refine biparts
1279 MB(52%)00:00:01                Iter   3   89.19%  Refine biparts
1279 MB(52%)00:00:01                Iter   3   91.89%  Refine biparts
1279 MB(52%)00:00:01                Iter   3   94.59%  Refine biparts
1279 MB(52%)00:00:01                Iter   3   97.30%  Refine biparts
1279 MB(52%)00:00:01                Iter   3  100.00%  Refine biparts
1279 MB(52%)00:00:01                Iter   3  102.70%  Refine biparts
1279 MB(52%)00:00:01                Iter   3  100.00%  Refine biparts
```

```
## Warning in file.remove(tempIn, tempOut): cannot remove file 'C:
## \Users\rosch\AppData\Local\Temp\RtmpKaqLdJ\file504c5abb3457.afa', reason
## 'Permission denied'
```

Phylogenetic Tree Creation

```
SeqLen <- as.numeric(lapply(UnknownDNAstring,length))
library(ggplot2)
qplot(SeqLen) + theme_bw() # results show that all sequences seem to have 100% similarity
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
KeepSeq <- SeqLen > 1000

UnknownSubset <- UnknownDNAstring[KeepSeq,]
UnknownSubAlign <- muscle(UnknownSubset, quiet = T)
```

```
## Warning in file.remove(tempIn, tempOut): cannot remove file 'C:
## \Users\rosch\AppData\Local\Temp\RtmpKaqLdJ\file504c7e016daf.afa', reason
## 'Permission denied'
```

```
UnknownSubAlignBin <- as.DNAbin(UnknownSubAlign)

UnknownDM <- dist.dna(UnknownSubAlignBin, model = "K80")


UnknownDMmat <- as.matrix(UnknownDM)

library(reshape2)
PDat <- melt(UnknownDMmat)
View(PDat)
```

```
UnknownTree <- nj(UnknownDM)

library(ggtree)
```

```
## ggtree v3.2.1  For help: https://yulab-smu.top/treedata-book/
##
## If you use ggtree in published research, please cite the most appropriate paper(s):
##
## 1. Guangchuang Yu. Using ggtree to visualize data on tree-like structures. Current Protocols
in Bioinformatics. 2020, 69:e96. doi:10.1002/cpbi.96
## 2. Guangchuang Yu, Tommy Tsan-Yuk Lam, Huachen Zhu, Yi Guan. Two methods for mapping and visu
alizing associated data on phylogeny using ggtree. Molecular Biology and Evolution. 2018, 35(1
2):3041-3043. doi:10.1093/molbev/msy194
## 3. Guangchuang Yu, David Smith, Huachen Zhu, Yi Guan, Tommy Tsan-Yuk Lam. ggtree: an R packag
e for visualization and annotation of phylogenetic trees with their covariates and other associa
ted data. Methods in Ecology and Evolution. 2017, 8(1):28-36. doi:10.1111/2041-210X.12628
```

```
##
## Attaching package: 'ggtree'
```

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

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

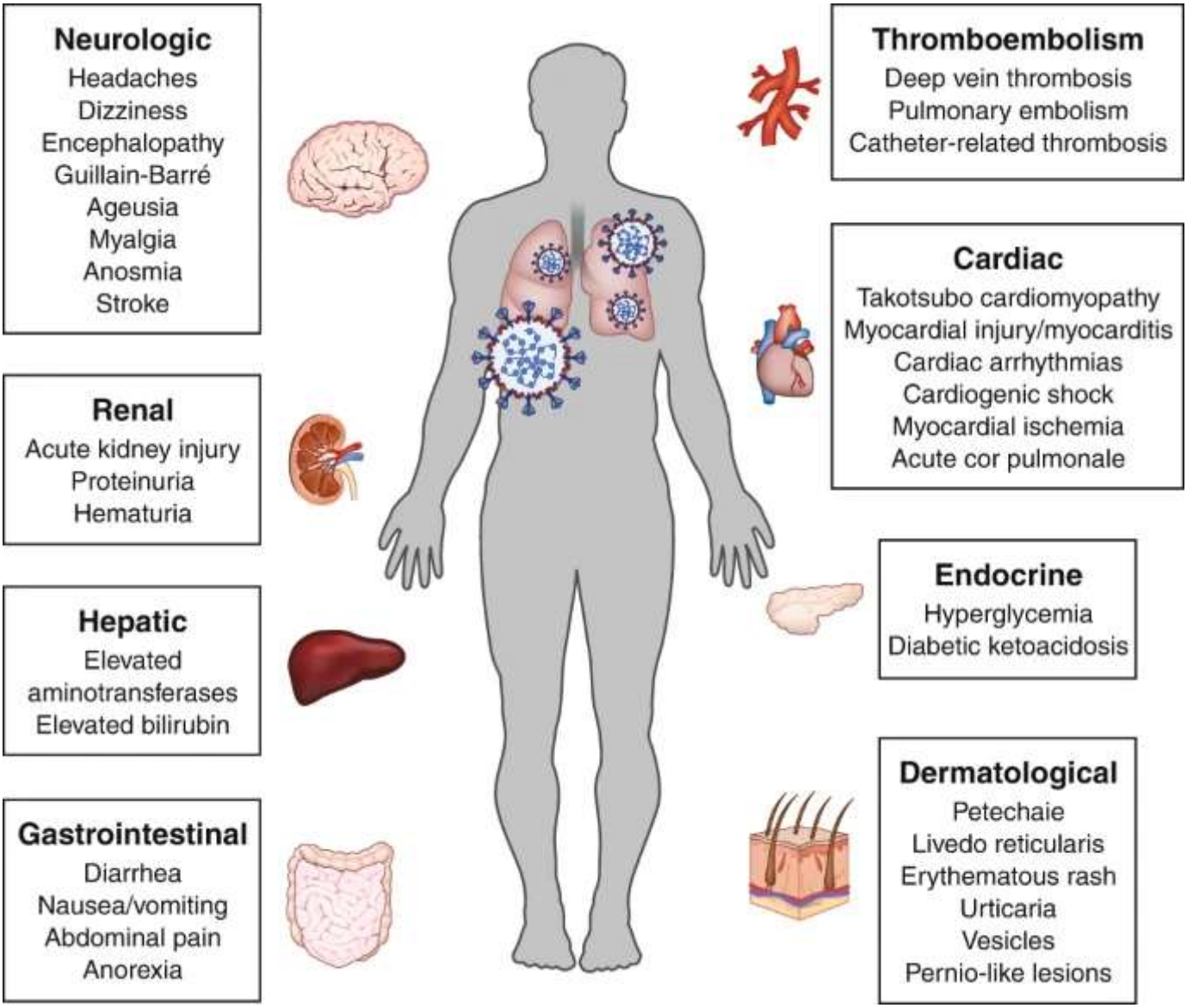
```
## The following object is masked from 'package:IRanges':
##
##     collapse
```

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

```
ggtree(UnknownTree)
```

Report

The unknown sequence that was identified from the patient is from the Coronavirus pathogen and IS something of concern, especially during current times as it is possible this individual may be positive for CoVID-19. As shown below, the Coronavirus pathogen has a detrimental long-term impacts on the human body, and early diagnosis and treatment of the virus is extremely beneficial. I am unsure why the phylogenetic tree is outputted as a single line; however, this may be due to the 100% similarity found between Coronavirus sequence data used to create the tree.



Impact of CoVID-19 pathogen on human body