732A51 Bioinformatics

LAB 2 Bioinformatics

Hugo Morvan
William Wiik

STIMA
Department of Computer and Information Science
Linköpings universitet

2024-11-20

Contents

1 Question 1

In this exercise you will perform statistical analysis of three nucleotide data sets. First download the sequences from GenBank and save them in a fasta le. For this use the provided R script, 732A51_BioinformaticsHT2023_Lab02_GenBankGetCode.R. This is a dataset of the RAG1 gene sequences from 33 lizard species. You are encouraged to read in detail the references in the script as they indicate many useful tools. Explore the dataset using the tools provided by the `ape` and `seqinr` packages. Take note of the lengths of all the sequences and the base composition.

```
## Gene bank accession numbers taken from http://www.jcsantosresearch.org/Class_2014_Spring_Comparative/
lizards_accession_numbers <- c("JF806202", "HM161150", "FJ356743", "JF806205",
                               "JQ073190", "GU457971", "FJ356741", "JF806207",
                               "JF806210", "AY662592", "AY662591", "FJ356748",
                               "JN112660", "AY662594", "JN112661", "HQ876437",
                               "HQ876434", "AY662590", "FJ356740", "JF806214",
                               "JQ073188", "FJ356749", "JQ073189", "JF806216",
                               "AY662598", "JN112653", "JF806204", "FJ356747",
                               "FJ356744", "HQ876440", "JN112651", "JF806215",
                               "JF806209")
lizards_sequences<-ape::read.GenBank(lizards_accession_numbers)
print(lizards_sequences)
```

```
## 33 DNA sequences in binary format stored in a list.
##
## Mean sequence length: 1982.879
##    Shortest sequence: 931
##     Longest sequence: 2920
##
## Labels:
## JF806202
## HM161150
## FJ356743
## JF806205
## JQ073190
## GU457971
## ...
##
## Base composition:
##     a     c     g     t
## 0.312 0.205 0.231 0.252
## (Total: 65.44 kb)
```

```
ape::write.dna(lizards_sequences, file ="lizard_seqs.fasta", format = "fasta",
               append =FALSE, nbcol = 6, colsep = " ", colw = 10)
```

1.1 Question 1.1

Simulate an artificial DNA sequence dataset. It should contain 33 sequence. The lengths of the sequences should be the same as in the lizard dataset, i.e. for each real sequence simulate an arti cial one. The simulation rule is as follows, each nucleotide is to be independently and randomly drawn from the distribution given by the base composition (frequencies) in the true lizard sequences. Save your dataset in a fasta format file. Remember to give unique names to your sequences. Report on the base composition in your simulated data.

Answer:

```
true_base_composition <- c(0.312, 0.205, 0.231, 0.252) # a c g t

# Length of each real sequence
sequence_lengths <- sapply(lizards_sequences, length)

# Simulating DNA
sim_sequences <- lapply(sequence_lengths, function(length) {
  rDNAbin(n = length, base.freq = true_base_composition)[[1]]
})

names(sim_sequences) <- paste0("seq_",1:33)

sim_sequences <- structure(sim_sequences, class = "DNAbin")

ape::write.dna(sim_sequences, file = "sim_lizard_seqs.fasta",
               format = "fasta", append = FALSE, nbcol = 6, colsep = " ", colw = 10)

sim_lizard_seqs <- read.FASTA("sim_lizard_seqs.fasta")
print(sim_lizard_seqs)
```

```
## 33 DNA sequences in binary format stored in a list.
##
## Mean sequence length: 1982.879
##    Shortest sequence: 931
##     Longest sequence: 2920
##
## Labels:
## seq_1
## seq_2
## seq_3
## seq_4
## seq_5
## seq_6
## ...
##
## Base composition:
##     a     c     g     t
## 0.312 0.203 0.230 0.255
## (Total: 65.44 kb)
```

a: 31.2% c: 20.3% g: 23.0% t: 25.5%

We obtain almost the same base composition as the true base composition for lizard sequences.

1.2 Question 1.2 *

2 Question 2

2.1 Question 2.1

Report some basic statistics on each sequence dataset: individual base composition, *GC* content, *CG*, *AT* content. Also translate your sequences into protein sequences (see Lab 1) and report on the amino acid composition. In your simulated sequences, how many times did you observe a stop codon inside your sequence? Does this occur in your true sequences? Comment.

```
true_sequences <- read.FASTA("lizard_seqs.fasta")
sim_sequences <- read.FASTA("sim_lizard_seqs.fasta")

calc_seq_stats <- function(all_seq, index_seq) {

 base <- base.freq(structure(all_seq[[index_seq]], class = "DNAbin"))
 GC_content <- base["g"] +  base["c"]
 AT_content <- 1 - GC_content

 base <- round(base, 3)


 print(paste0("Base composition for ", names(all_seq)[index_seq],
              ": a=",base["a"],", c=",base["c"],", g=",base["g"],", t=",base["t"]))

 print(paste0("GC content for ", names(all_seq)[index_seq],
              ": ",round(GC_content,3)))

 print(paste0("AT content for ", names(all_seq)[index_seq],
              ": ",round(AT_content,3)))

}

# True lizard sequences
calc_seq_stats(all_seq = true_sequences, index_seq = 1)
```

```
## [1] "Base composition for JF806202: a=0.29, c=0.203, g=0.244, t=0.264"
## [1] "GC content for JF806202: 0.446"
## [1] "AT content for JF806202: 0.554"
```

```
# True lizard sequence 1
calc_seq_stats(all_seq = true_sequences, index_seq = 1)
```

```
## [1] "Base composition for JF806202: a=0.29, c=0.203, g=0.244, t=0.264"
## [1] "GC content for JF806202: 0.446"
## [1] "AT content for JF806202: 0.554"
```

```r
# Simulate lizard sequence 1
calc_seq_stats(all_seq = sim_sequences, index_seq = 1)
```

```
## [1] "Base composition for seq_1: a=0.286, c=0.207, g=0.257, t=0.251"
## [1] "GC content for seq_1: 0.464"
## [1] "AT content for seq_1: 0.536"
```

```r
# ==============================================================================
# From chatGPT, how to get frequency every charchater per sequence for an out file
# ==============================================================================


process_protein_file <- function(file_path) {
  # Read the file
  lines <- readLines(file_path)

  # Extract headers (lines starting with ">")
  headers <- grep("^>", lines, value = TRUE)

  # Extract sequence lines (lines not starting with ">")
  header_indices <- grep("^>", lines)

  # Create a list of sequences
  sequences <- mapply(function(start, end) {
    paste(lines[(start + 1):(end - 1)], collapse = "")
  }, header_indices, c(header_indices[-1] - 1, length(lines)), SIMPLIFY = TRUE)

  # Remove ">" from headers
  headers <- gsub("^>", "", headers)

  # Create the data frame
  protein_data <- data.frame(
    Header = headers,
    Sequence = sequences,
    stringsAsFactors = FALSE
  )


}
```

```r
# https://www.ebi.ac.uk/jdispatcher/st/emboss_transeq/summary?jobId=emboss_transeq-I20241120-223345-0023
transeq_true <- process_protein_file("transeq_true.out")
chars <- strsplit(transeq_true[1,2], NULL)[[1]] # JF806202
char_table <- prop.table(table(chars))
round(char_table[order(table(chars))],2)
```

```
## chars
##    X    Y    D    H    F    G    E    N    C    I    W    P    A    V    *    R
```

```
## 0.00 0.01 0.02 0.02 0.02 0.02 0.03 0.03 0.03 0.04 0.04 0.05 0.05 0.05 0.05 0.05
##    T    Q    K    M    S    L
## 0.05 0.06 0.06 0.07 0.09 0.15
```

```
# https://www.ebi.ac.uk/jdispatcher/st/emboss_transeq/summary?jobId=emboss_transeq-I20241120-221152-0336
transeq_sim  <- process_protein_file("transeq_sim.out")
chars <- strsplit(transeq_sim[1,2], NULL)[[1]] # seq_1
char_table <- prop.table(table(chars))
round(char_table[order(table(chars))],2)
```

```
## chars
##    F    W    C    Y    M    H    D    N    P    *    E    K    T    I    V    A
## 0.02 0.02 0.02 0.02 0.03 0.03 0.04 0.04 0.04 0.04 0.04 0.05 0.05 0.06 0.06 0.06
##    Q    G    S    R    L
## 0.06 0.07 0.08 0.08 0.09
```

2.2 Question 2.2*

2.3 Question 2.3

Align your sequences using software of your choice (a starter for R: https://stackoverflow.com/questions/4497747/how-to-perform-basic-multiple-sequence-alignments-in-r, you can also look what `Biopython`, `BioPerl` offer, use the `Clustal` family of programs or something else of your choice). Choose a distance measure between sequences, calculate for each alignment the distances between all pairs of sequences. Then, plot heatmaps visualizing the distances. Comment on what you can observe

3 Question 3

3.1 Question 3.1

Construct (using algorithm and software of your choice) phylogenetic trees from the three mul tiple alignments (or distance matrices) done in Question 2.3. You might want to look at the functions o ered by `ape`, `phangorn` (https://cran.r-project.org/web/packages/phangorn/vignettes/Trees.pdf) or go for some completely di erent software. Plot the inferred trees. Are the two based on the simulated data similar to expected? Perform a phylogenetic bootstrap analysis and report the bootstrap support for the individual clades, you can look at `ape::boot.phylo()`.

3.2 Question 3.2*