Stat 610 Homework 2

Due Wednesday, September 18, 11:59pm

Assignment

In this assignment, you will get some practice with regular expressions. It is inspired by a real problem in immunology which is too complicated to describe here but which you can talk to me about if you are interested.

As in the previous homework, please make a first attempt without using generative AI. If you are happy with the code you wrote on your own, you can turn in just what you wrote. If you felt like you needed to use chatGPT or something similar, compare the code you wrote to what chatGPT gives you in response to the questions. Note whether the code works and whether there are any stylistic or substantive differences with what you wrote.

You should do the following:

- Download the file `ighv.fasta` from the course website and read it in to R using `ighv <- read.delim('path/to/ighv.fasta', header = FALSE, stringsAsFactors = FALSE)`, where you replace `'path/to/ighv.fasta'` with the location where the file lives on your computer. What type of object is `ighv`? What happens if you use `header = TRUE` instead of `header = FALSE`?

- `ighv.fasta` is a fasta file, which you can read about here. A fasta file represents a set of sequences. Each sequence in the fasta file has two corresponding lines: a description line, which starts with > and is followed by a description of the sequence, and a sequence line, which contains the AGCT nucleotide representation of the sequence.

  We will want to work with the descriptions and the nucleotide sequences separately, so we need to separate out the descriptions from the sequences. Using a for loop or the vector subsetting techniques from class, create one vector that contains just the sequence descriptions and one vector that contains just the nucleotide representations of the sequences.

- Our next task is to check that the sequences were extracted correctly. Make a regular expression that checks whether there are any letters aside from A, G, C, or T (the only valid letters for a nucleotide sequence). Apply this to your vector of sequences to check whether the sequences you extracted are valid.

- Next, we check that the descriptions were extracted correctly. The descriptions should all be of the form `>IGHV`, then a digit, then a `-`, then one or two digits, then $*$, then two digits. Develop a regular expression that corresponds to a valid description line and use it to check that all of the elements of the description vector are valid.

- Make a regular expression that matches strings of the form A or T followed by C or G followed by A or T.

- Modify the regular expression above so that it matches strings with at least five A/T followed by C/G followed by at least five A/Ts. Search for the regular expression in your sequence vector using the `grep` function. How many sequences have that particular pattern?

- Using `grep` above checked whether there were *any* instances of the specified pattern in each sequence. We would also like to know whether there are any sequences for which the pattern occurs more than once. Modify the code from the previous question to use `gregexpr` instead of `regexpr`. What is the type of output that it gives? What does each element of the output represent? Finally, check whether any of the sequences have more than one instance of the specified pattern.

Submission parameters

You should submit an Rmd file and the corresponding pdf or html on canvas. The files should contain both the code you ran and the answers to the problems.