NGS for protein variants - exercise 2

Goal: parse real-world mutagenesis data (including clean-up) and calculate log odds ratios

Background in https://www.biorxiv.org/content/10.1101/274753v1.full

- ❏ Read in the "dim" and "bright" datasets as uploaded to Absalon. This file is tab separated, not in fasta format, so you may have to switch the fasta reader to a csv/table reader. If using R, the flag `stringsAsFactors=F` is recommended for reading. Initially, process each of them separately. We will only combine them at the log odds stage.
    - ❏ During code development, it may be a good idea to only work with the first 10-20 lines to reduce run times. Note that the `Biostrings translate()` function is not particularly fast and will need several minutes if applied to thousands of sequences.
        - ❏ Therefore it may be a good idea to store the translated sequences in a file and, for start from that file for development of downstream analyses (e.g. matrices analysing and summarising the differences between the protein sequences)
    - ❏ Some issues (such as short or rare sequences) might only pop up with sequences that appear later in the file, so, not all debugging is done if it works for the first 10.
- ❏ The reference (wild-type) DNA is the same native_DNA.fa as in Exercise 1.
- ❏ The first column in each file is the count how often this DNA sequence has been observed. We will only use it for filtering out very rare sequences (you'll get to determine what exactly the threshold is yourselves). Technical aspects like PCR amplification can distort this number, so we will not use it for interpretation of variant fitness.
- ❏ Each of these sequences is a variant. Problems like sequencing errors mean that there might be more unique sequences than actual unique variants in the sample. Again, discarding what was seen rarely is a good idea to remove uncertain variants.
- ❏ Data clean-up
    - ❏ Omit sequences that have gaps - these typically lead to frameshifts
    - ❏ Similarly, remove sequences that are shorter than the native DNA sequence
    - ❏ You can also choose to omit sequences that have a different length than the wild type for easier comparison
    - ❏ Check for DNA sequences that contain non-DNA letters
    - ❏ Omit sequences with premature STOP, like before
- ❏ If you want to use `lapply()` for translation, try a construct like
  ```
  ngs_data$protein <- as.character(lapply(ngs_data$DNA,
  function(s) toString(translate(DNAString(s)))))
  ```
- ❏ Determine differences of the variant sequences to the wild-type sequence **in protein space** just like you did in Exercise 1. You will need both the identity of the change

(which WT for which new amino acid) and its position in the sequence. Create a 20x20 substitution matrix for each of the two datasets. Then calculate the log-odds ratios for each available datapoint to compare which substitutions are enriched in the dim or the bright dataset. Given your biochemical knowledge, are these expected?

- ❏ Next we will create mutation-by-position maps, like in Fig. 2 in the paper. If you haven't done so before Iidentify missense variants again, but now store them in a format that keeps track of both the position and the target (=new) amino acid.
- ❏ Use the mutation-by-position maps from the dim and bright datasets to calculate log odds ratios for each variant at each position where data is available
    - ❏ The main limitation here is the lack of data for the same position and variant in both datasets, a side effect of random mutagenesis. You can supplement the variants not observed in the dim set with pseudocounts if you like.