Antibody Sequence Analysis

Sequence Logos from Multiple Sequence Alignment,
Humanization Models

Enrico Frigoli, Lotte Draijer, Clara Canavese, Charlotte Resch

University of
Southern Denmark

Introduction to Bioinformatics
Prof. Konrad Krawczyk
Fall 2022

Github: https://github.com/enricofrigoli/bioinformatics_fall22.git

Contents

1 Introduction

Antibodies are proteins produced by the immune system to respond to the presence of foreign substances. Antibodies are able to recognize unique parts on such substances, called antigens, and bind to those.

The basic structure of an antibody in mammals consists of two pairs of polypeptide chains that form a Y-shape [1]. The polypeptide chains can be distinguished in two identical heavy chains and two identical light chains, connected by disulfide bonds. Each chain is composed by a series of domains, that can be variable (V) or constant (C). The sub-region, called F_V, that binds to the antigen is composed of the variable domain of the heavy chain and the variable domain of the light chain [2].

Each variable domain is usually subdivided into four Framework Regions (FRs) separated by three Complementary Determining Regions (CDRs), also called hypervariable regions. When the protein folds, the three CDRs come together on the surface of the antibody and create an antigen-binding site (together with the 3 CDRs of the other chain).

Thus, CDRs differ significantly from antibody to antibody, since they are responsible for the affinity towards only one specific epitope, which is a distinct section of the antigen surface that is recognized by a specific antibody (different antibodies can recognized different epitopes of one antigen). Instead, the FRs serve as scaffold to hold the CDRs in position, thus they are expected to be more conserved across the range of all antibodies of a given species.

In the development of therapeutic monoclonal antibodies, the starting point is often an immunized non-human organism, i.e. a mouse, which antibodies are collected and engineered to make them "more human", thus lowering the probability of having an anti-mouse response following administration.

In this project, we explore the sequence variability of antibodies and the use of different tools to distinguish between mouse and human antibodies, in order to aid the humanization effort. The full code is available in the GitHub repository.

2 Sequence Logos from Multiple Sequence Alignment

The input files are two FASTA files containing 9997 protein sequences each, representing the variable domain of the heavy chain of human and mouse antibodies respectively.

Thanks to the regular subdivision into CDRs and FRs of the sequences taken into consideration, it is possible to perform Multiple Sequence Alignment (MSA) to spot FRs segments and CDRs in the given sequences by looking at the variability of amino acid residues, given that FRs are expected to align together.

2.1 Multiple Sequence Alignment

To perform MSA we used both Clustal Omega [3] and MUSCLE [4]. We downloaded Clustal Omega locally using `conda` and ran it with the Biopython wrapper. MUSCLE was also used locally by downloading the executable file and running the alignment via command line. We used the Super5 algorithm (`-super5` argument) that allows to align large datasets more efficiently.

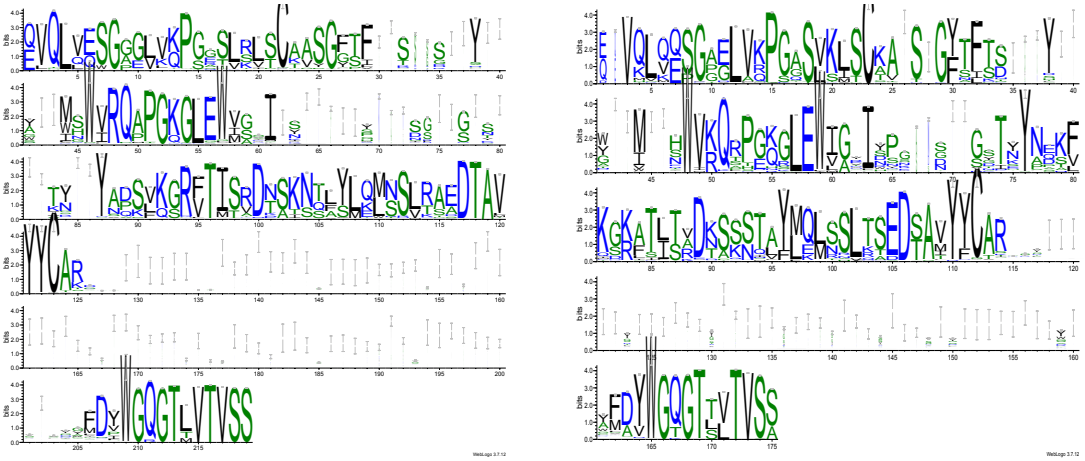
2.2 Sequence Logo

In sequence logos, the overall height of the stack indicates the *information content* of the column, i.e., its relative entropy distance from an assumed background distribution, which

measures the conservation of a position in a profile. The height of symbols within the stack indicates the relative frequency of each amino acid at that position.

We used the output files from the MSA to generate the sequence logos. We installed WebLogo using `pip`, then we wrote a bash script that automatically creates the sequence logos from previously generated FASTA files.

In both human and mouse sequence logos (Figures 1, S1), we can see more conserved regions alternating with more variable regions. The former corresponds to the framework: a higher level of self information means lower entropy and so more conserved positions; the latter corresponds to the CDRs, that are the most variable parts of antibodies where the entropy is higher and positions are less conserved. As expected, there are 3 non-consecutive CDRs regions in both the human and the mouse variable region of the antibodies.



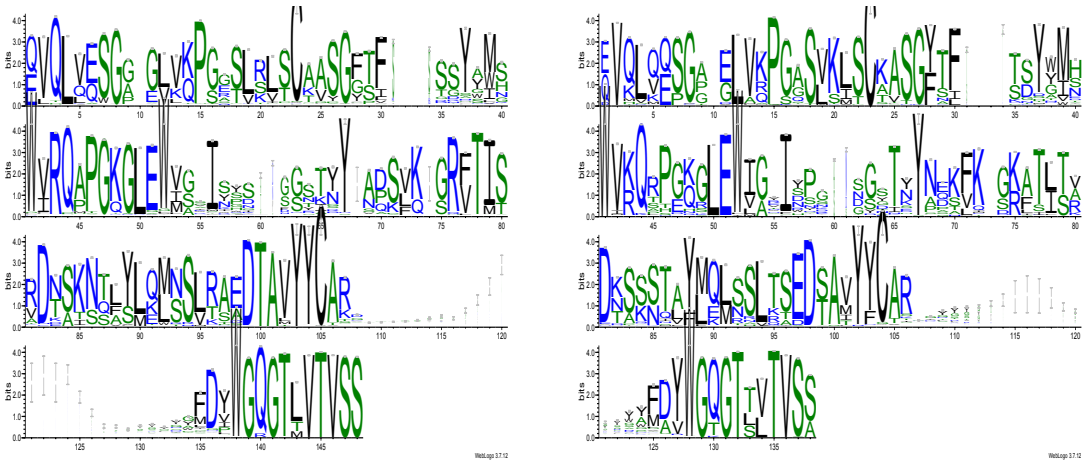
(a) Human (b) Mouse

Figure 1: Sequence logos for human and mouse antibody sequences aligned with MUSCLE. The positions without letters are the result of a large percentage of gaps in a specific position across the MSA, thus, such positions should be ignored.

2.3 Antibody Numbering

To perform *antibody numbering* using the IMGT numbering scheme [5] we installed the ANARCI package [6] via `conda`. We used the `run_anarci` wrapper to ensure parallel computation and direct FASTA parsing. The alignment was restricted to human or mouse germline only, depending on the dataset. After that, we created a Python function that takes the numbering CSV file as input and gives as output a FASTA file containing the aligned gapped sequences. Finally, we created the Sequence Logos for both the human and the mouse sequences using WebLogo, as done in the previous section. In these logos (Figure 2), positions without letters correspond to positions in the alignment in which there is a high frequency of gaps.

The lengths of the sequence logos obtained with the first Multiple Sequence Alignment are greater than those obtained through HMM and antibody numbering with IMGT. That is because `anarci` aligns each sequence to a set of Hidden Markov Models (HMMs) using HMMER3. Each HMM describes the putative germ-line sequences for a domain type (VH, Vk or Vj, Va or Vb) of a particular species. The most significant alignment classifies the domain type and the alignment is then translated into a chosen numbering scheme (IMGT in our case). In other words, the second method is imposing some preconceived assumptions, while the first method is not, so the length is left free to vary more.



(a) Human (b) Mouse

Figure 2: Sequence logos obtained from the alignments extracted from the antibody numbering. Human CDRs span the following positions in the plot: 26-37, 55-64, 105-136. Mouse CDRs: 26-37, 55-64, 104-126. Positions without letter are occupied mostly by gaps across the set of sequences.

Moreover, thanks to the IMGT numbering, we were able to identify the domain of our sequences: it is the variable part of the heavy chain of the antibody (VH).

Another advantage of using `anarci` is that the CDRs are much easier to identify because their position in the sequence can simply be retrieved from the numbering. In fact, all the sequences in the alignment have the same length; this is achieve in ANARCI by adding gaps into CDRs to compensate for the variability in length from CDRs of different antibodies. Thus, the positions in the alignment (and in the plot) does not correspond to positions in the numbering, since the numbering makes use of a convoluted notation for CDRs of different length (i.e. 111A, 111B, etc).

3 Antibodies humanization

3.1 Humanization tool in Python

We developed in Python the tool that outputs whether the input sequence has higher identity to mice or human sequences and finds heavy chain sequences of therapeutic antibodies (Muromonab, Bevacizumab, Caplacizumab). The code can be found in Appendix A.2 and on GitHub. We used an algorithm performing local alignment which finds the subsequences that align the best. Local alignment (`pairwise2.align.localxx(seq1,seq2)`) is used as it is more suitable for more divergent sequences or distantly related sequences. We set the match parameter to x, so identical characters have a score of *1* otherwise the score is *0*. The chosen gap penalty parameter is also x, meaning there are no gap penalties.

3.2 Identifying divergent positions

A more efficient way to aid the humanization effort would be to spot positions in the query sequence that do not align well to human sequences and suggest them as a potential target for engineering.

To do so, the numbering obtained with ANARCI was used, since it produces gapped sequences of the same length in which CDRs and FRs are aligned according to the IMGT scheme, thus allowing position-wise residue comparison between sequences.

The query sequences (Muromonab, Bevacizumab, Caplacizumab) and the dataset of human sequences were concatenated in a single FASTA file that was given as input to ANARCI. The alignment was then extracted in the same way as done in the previous section, and two FASTA files were produced, one with the query sequences and one with the sequences from the dataset.

Since all the sequences have the same length, it is possible to compare position by position the residues in the query sequences with the ones in the dataset sequences. Thus, it is possible to define a measure that indicates how much each residue in the query sequence is conserved across the human sequences in order to identify regions that do not align well, which might be potential target for engineering. The position-wise frequency of each residue across the human dataset was chosen as such measure.

In order to quickly compute the frequencies for each query sequence, a Position Frequency Matrix was computed from the dataset of human sequences; gap frequency was also included. During the computation of the frequencies, the positions in the query corresponding to a gap (-) were discarded (Figure 3). The frequency was obtained dividing the counts by the number of sequences in the dataset. The frequency of gaps was plotted in order to provide a visual intuition about regions in which a low frequency in the query sequences is due to a large amount of gap in specific positions in the dataset sequences; that scenario is especially true deeply into CDRs regions, which were highlighted in yellow.

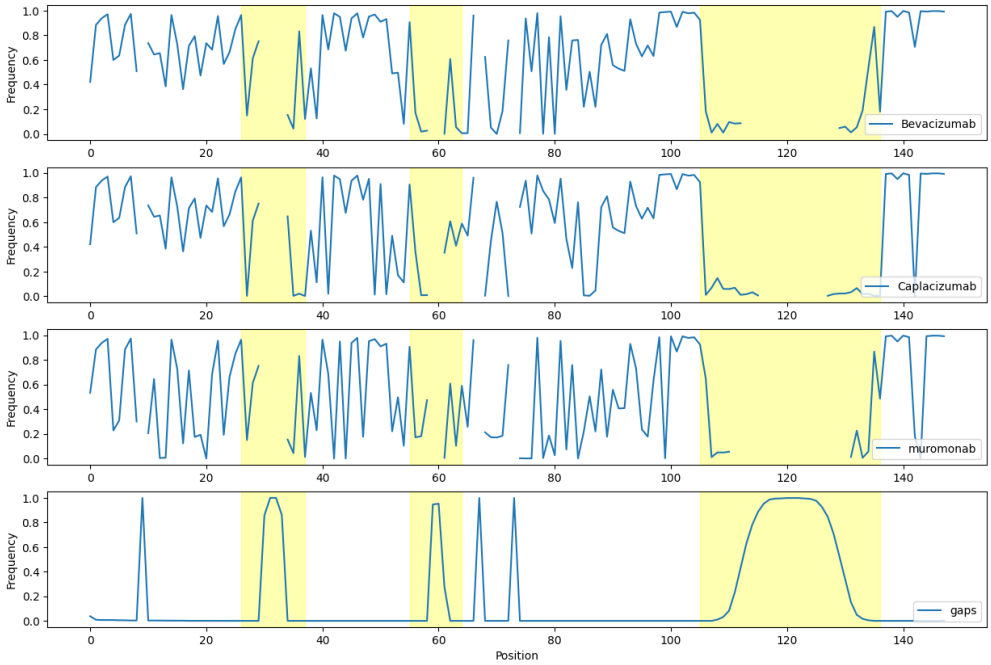


Figure 3: Frequency of each residue found in the query sequence across the dataset of human sequences. Position covering CDRs are highlighted in yellow. The position-wise frequency of gaps in the dataset is also plotted to highlight regions in which a low frequency in the query sequences is due to a large amount of gaps in that position in the alignment.

As expected, the frequency suddenly drops in CDRs, while is higher in framework regions. Moreover, it is possible to spot a similarity between Bevacizumab, Caplacizumab and Muromomab in terms of the frequencies of some residues in the frameworks. This is in accordance with the fact that residues in frameworks are generally more conserved as their functions is more structural, whereas residues in the CDRs are much less conserved since they confer affinity and specificity to a given antibody. For this reason, in theory,

only positions in the framework regions can be targets for engineering, since modifying the CDRs would cause the antibody to lose its function.

3.3 Quantification of antibody nativeness using LSTM network

A more advanced way of quantifying the nativeness of a given sequence makes use of a Long Short-Term Memory (LSTM) network published in [7]. The underlying assumption is that, given an input sequence, a well-trained model should be able to predict any residue by learning information from its neighbors. The output of the model is the averaged sum of negative logarithms of all conditional probabilities; so lower scores indicate a higher degree of nativeness. We trained the model on the dataset of human sequences, setting the embedding layer dimension to 128 and the hidden layer dimension to 128; other (self-explanatory) parameters we used were `gapped=True` and `fixed_len=True`. We plotted the frequencies of scores obtained after evaluating 20% of the total number of human and mouse sequences [Figure 4]. As expected, human sequences score shows a peak on a lower LSTM score compared to the mouse sequences score. Also, the three therapeutic antibodies scored accordingly (Table 1).

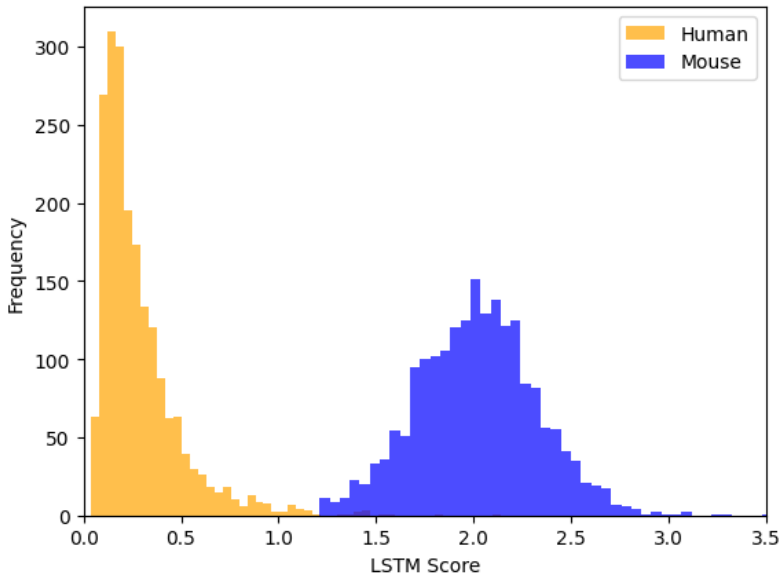


Figure 4: Distribution of sequence scores of human and mouse sequences obtained with the LSTM model trained on human sequences

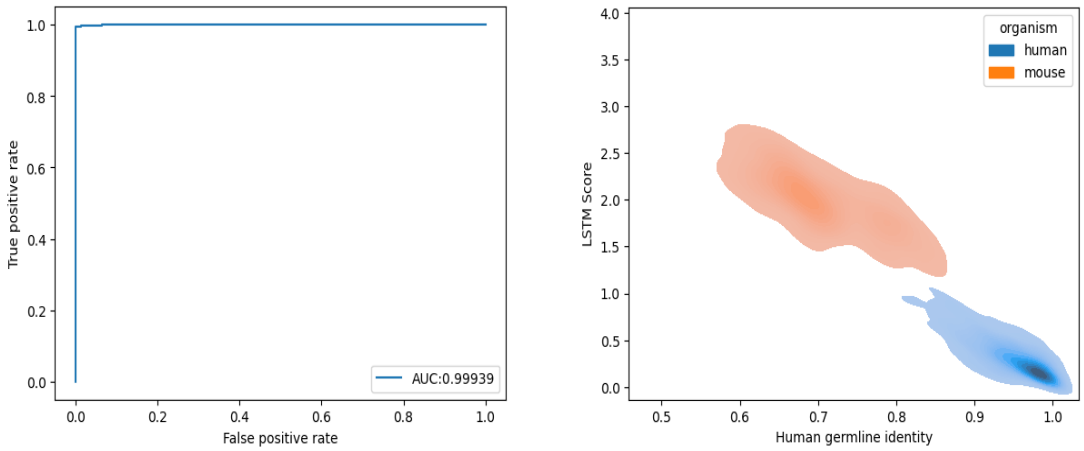
We plotted the Receiver Operating Characteristic (ROC) curve where positives correspond to human and negatives to mouse [Figure 5a]. The ROC analysis shows that in classifying sequences as human-like, the model has an AUC of 0.99939 for mouse sequences.

Finally, we plotted the LSTM score of human and mouses sequences versus the human germline V gene identity as a Kernel Density Estimate (KDE) plot [Figure 5b]. This identity score corresponds to the sequence identity over the v-region to the most sequence identical germline, and it was extracted from the ANARCI numbering. As expected, we found the LSTM score to be correlated to the sequence identity of the closest human germline v-gene: the higher the identity, the lower the LSTM score. By looking at this plot, it is evident that most of the human sequences in the human dataset had a germline identity close to 1; so, the LSTM model was trained on sequences closely related to the germline sequence.

Antibody	LSTM score	V identity
Bevacizumab	1.50	0.77
Caplacizumab	1.72	0.82
Muromonab	1.83	0.72

Table 1: LSTM score and V gene identity of commercially available sequences. As expected, humanized antibodies (Bevacizumab and Caplacizumab) score lower than Muromomab, which is from mouse but not humanized. If plotted in the KDE plot, all the three point lie within the mouse area.

Thus, a correlation between LSTM score and germline identity is expected a priori, since sequences with lower germline identity are expected to score higher in the evaluation simply because they are less prevalent in the training dataset. However, this correlation was maintained in mouse sequences as well, pointing to the fact that the LSTM model is able to model the sequences in a biologically relevant way. Finally, our results were in line with the results of [7].



(a) ROC curve

(b) Kernel Density Estimation

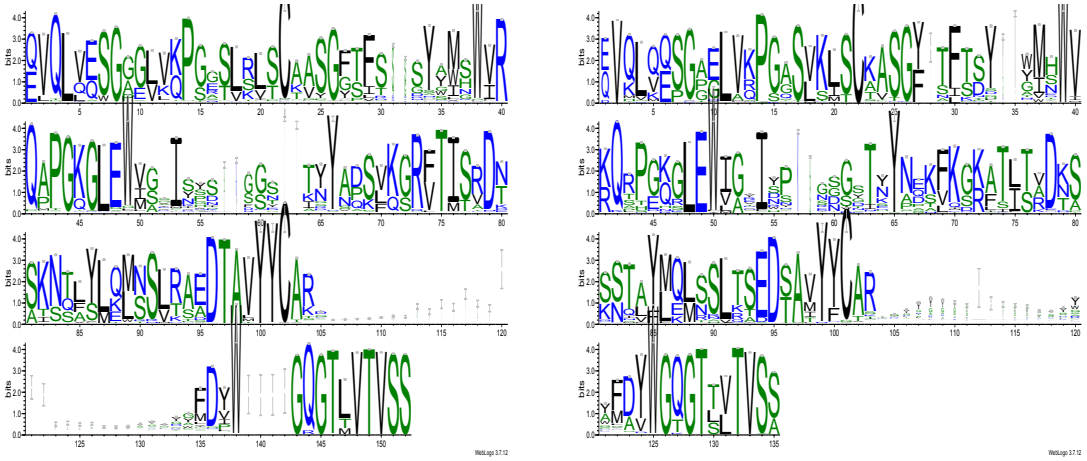
Figure 5: **(a)** ROC curve showing the performance of the LSTM model trained on human sequences in distinguishing human and mouse sequences. **(b)** Kernel Density Estimation plot showing the relationship between LSTM score and the identity to the V-gene computed by ANARCI.

4 Conclusion

In this analysis of antibody sequences, we successfully performed Multiple Sequence Alingment using different tools, we performed Antibody Numbering using ANARCI, and we explored different ways to compute the similarity between a given sequence and a dataset of human sequences to potentially aid the humanizaton effort.

A Appendix

A.1 Supplementary Figures



(a) Human (b) Mouse

Figure S1: Sequence logos for human and mouse antibody sequences aligned with Clustal Omega

A.2 Code for humanization tool

```
import Bio.SeqIO as SeqIO
from Bio import pairwise2
import os
from datetime import datetime

# get the directories
# used files have to be stored in same directory as the code
cwd = os.getcwd()
humDir = os.path.join(cwd,"human.fa")
mouseDir = os.path.join(cwd, "mouse.fa")
muromonab = os.path.join(cwd, "muromonab.FASTA")
bevacizumab = os.path.join(cwd, "bevacizumab.FASTA")
caplacizumab = os.path.join(cwd, "caplacizumab.FASTA")

# get muromonab sequence
muromonab = SeqIO.to_dict(SeqIO.parse(muromonab, "FASTA"))
muromonab = list(muromonab.values())
muromonab = muromonab[0].seq

# get bevacizumab sequence
bevacizumab = SeqIO.to_dict(SeqIO.parse(bevacizumab, "FASTA"))
bevacizumab = list(bevacizumab.values())
bevacizumab = bevacizumab[0].seq

# get caplacizumab sequence
caplacizumab = SeqIO.to_dict(SeqIO.parse(caplacizumab, "FASTA"))
caplacizumab = list(caplacizumab.values())
caplacizumab = caplacizumab[0].seq


# local alignment of given sequence with all sequences given sequences of humans and mice
```

```
# use global alignment if sequences have roughly same length else use local
def alignment(seq,humDir,mouseDir):
    print(datetime.now())

    # get human and mouse dicts
    humDict = SeqIO.to_dict(SeqIO.parse(humDir, "FASTA"))
    mouseDict = SeqIO.to_dict(SeqIO.parse(mouseDir, "FASTA"))

    # get sequences
    humSequences = [s.seq for s in humDict.values()]
    mouseSequences = [s.seq for s in mouseDict.values()]

    # start alignment for all given human sequences
    print('starting pairwise alignment human dataset\n')
    humAlignments = [pairwise2.align.localxx(seq,humS) for humS in humSequences]
    print("done pairwise alignment human dataset\n")
    print(datetime.now())

    # start alignment for all given mouse sequences
    print('starting pairwise alignment mouse dataset\n')
    mouseAlignments = [pairwise2.align.localxx(seq,mouseS) for mouseS in mouseSequences]
    print("done pairwise alignment mouse dataset\n")
    print(datetime.now())

    return humAlignments, mouseAlignments

# find the highest score of all possible alignments
def get_highest_score(humAlignments,mouseAlignments):

    hMax = 0
    mMax = 0

    # find highest score of human alignments and highest score of mouse alignments
    for hA,mA in zip(humAlignments,mouseAlignments):
        for align in hA:
            if align.score > hMax:
                hMax = align.score
                hSeq = align.seqB
        for align in mA:
            if align.score > mMax:
                mMax = align.score
                mSeq = align.seqB

    # compare human and mouse highscore and return highest
    if hMax > mMax:
        return {'dataset': 'human', 'seq': hSeq, 'score': hMax}
    return {'dataset': 'mouse', 'seq': mSeq, 'score': mMax}


hMuromonabSeq, mMuromonabSeq = alignment(muromonab, humDir, mouseDir)
highMuromonabSeq = get_highest_score(hMuromonabSeq, mMuromonabSeq)
print("highest score for alignment with muromonab: "+ str(highMuromonabSeq))

hBevacizumabSeq, mBevacizumabSeq = alignment(bevacizumab, humDir, mouseDir)
highBevacizumabSeq = get_highest_score(hBevacizumabSeq, mBevacizumabSeq)
print("highest score for alignment with bevacizumab: " + str(highBevacizumabSeq))

hCaplacizumabSeq, mCaplacizumabSeq = alignment(caplacizumab, humDir, mouseDir)
highCaplacizumabSeq = get_highest_score(hCaplacizumabSeq, mCaplacizumabSeq)
print("highest score for alignment with caplacizumab: " + str(highCaplacizumabSeq))
```

References

[1] The Editors of Encyclopaedia Britannica. *antibody*. Sept. 2022. URL: https://www.britannica.com/science/antibody.

[2] Janeway Jr C. A., Travers P., and Walport M. *Immunobiology: The Immune System in Health and Disease*. 5th ed. Garland Science, 2001.

[3] Fabian Sievers and Desmond G. Higgins. "Clustal Omega for making accurate alignments of many protein sequences". In: *Protein Science* 27.1 (2018), pp. 135–145. DOI: https://doi.org/10.1002/pro.3290. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/pro.3290. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/pro.3290.

[4] Robert C Edgar. "MUSCLE v5 enables improved estimates of phylogenetic tree confidence by ensemble bootstrapping". In: *bioRxiv* (2021).

[5] M P Lefranc. "Unique database numbering system for immunogenetic analysis". In: *Immunology Today* 18 (1997). DOI: 10.1016/S0167-5699(97)01163-8.

[6] James Dunbar and Charlotte M. Deane. "ANARCI: antigen receptor numbering and receptor classification". In: *Bioinformatics* 32.2 (Sept. 2015), pp. 298–300. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btv552. eprint: https://academic.oup.com/bioinformatics/article-pdf/32/2/298/6688939/btv552.pdf. URL: https://doi.org/10.1093/bioinformatics/btv552.

[7] Andrew M Wollacott et al. "Quantifying the nativeness of antibody sequences using long short-term memory networks". In: *Protein Engineering, Design and Selection* 32.7 (Aug. 2019), pp. 347–354. ISSN: 1741-0126. DOI: 10.1093/protein/gzz031. eprint: https://academic.oup.com/peds/article-pdf/32/7/347/33487423/gzz031.pdf. URL: https://doi.org/10.1093/protein/gzz031.