Biostatistics (STA5195) Lab 2 Report

Turning Biology into Mathematics

Christopher Rutherford

August 31, 2020

Abstract

In this lab, we will pull some data from the uniprot database, which contains amino acid sequences of many kinds of proteins. We will use 2000 antibody proteins and 2000 non-antibody proteins for this analysis. Each sequence is represented by a string of letters, with each letter representing an amino acid. We found that the distribution of amino acids in a protein is somewhat uniform, but some acids are much more common than others. Additionally, the length of the proteins' sequence is somewhat normally distributed, but has a very strong right skew due to a small amount of very long sequences.

1 Introduction

Amino acids are the basic building blocks of proteins. We represent each amino acid in the sequence with a unique letter, which are set by the commission for biochemical nomenclature of the IUPAC. We will be using Python to pull the sequences for these proteins from the UniProt database, which contains the amino acid sequence for many proteins. In particular, we are interested in converting these sequences into a more machine- and human-readable format. Since each sequence is represented by a string of letters, we can map these letters to numbers to make the data easier to plot and analyze.

2 Theory

Because the original data comes in a sequence of letters, we need to convert these letters to numbers for convenience. In order to convert our sequence of letters to numbers, we must map the given letters to their corresponding numbers. For convenience, we will go in alphabetical order, starting with mapping a to 0. Using Python's built-in `ord` function allows us to obtain the decimal representation of each letter by converting its hexadecimal unicode value to a decimal number. The letter a is represented by unicode value 0061_{16}, which is 97_{10}. As such, we can subtract 97 from each letter's decimal number value to get an interval of [0, 25] for our new numbers. We can represent this conversion with the following equation:

$$n = ord(x) - 97 \tag{1}$$

where n is the number after conversion, and x is the letter from the sequence.

3 Procedures

We begin by importing the required libraries for our import and analysis. We will be using Python to import the proteins' amino acid sequences directly from the Uniprot database.

```
!pip install git+https://github.com/williamedwardhahn/mpcr
from mpcr import *
```

This library provides us with the necessary function to import the data, along with other common analysis libraries, such as numpy and matplotlib. Now we can import the data.

```
# This code will create a dataset from the uniprot database
X, Y = get_uniprot_data('=antibody', '!antibody', 2000)
# X contains the antibody proteins
# Y contains the non-antibody proteins
```

This line of code creates an array of 2000 antibody-related proteins saved in `X`, and another 2000 non-antibody-related proteins saved in `Y`. Now that we have the data, we perform a basic inspection to ensure the data was imported properly:

```
number_X = len(X) # set number_X to the length of X
number_Y = len(Y) # set number_Y to the length of Y
print(number_X)
print(number_Y)
```

```
2000 2000
```

This confirms that we imported the desired amount of proteins in each variable. We can also take a look at what one of these sequences looks like:

```
X[0] #Amino acid sequence of the first protein on the list of
   proteins associated with 'antibody'
```

Output:

mvllrvlilllswvaglggqygnplnkyirhyeglsydvdslhqkhqrakravshedqflrldfhahgrhfnlrmkrdtslfse
efrvetsnavldydtshiytghiygeegsfshgsvidgrfegfiqthggtfyvepaeryikdrtlpfhsviyheddikyphkygpqg
gcadhsvfermrkyqmtgveevtqtpqekhaingpellrkkrttvaekntcqlyiqtdhlffkyygtreaviaqisshvkaidtiy
qttdfsgirnisfmvkririnttadekdptnpfrfpnigvekflelnseqnhddyclayvftdrdfddgvlglawvgapsgssggic
eksklysdgkkkslntgiitvqnygshvppkvshitfahevghnfgsphdsgtectpgesknlgqkengnyimyaratsgdkln
nnkfslcsirnisqvlekkrnncfvesgqpicgngmveqgeecdcgysdqckdeccydanqpegkkcklkpgkqcspsqgpcct
ahcafksktekcrddsdcakegicngitalcpasdpkpnftdcnrhtqvcingqcagsicekhgleectcassdgkddkelchvcc
mkkmepstcastgsvqwnkyflgrtitlqpgspcndfrgycdvfmrcrlvdadgplarlkkaifspelyeniaewivaywwavl
lmgialimlmagfikicsvhtpssnpklpppkplpgtlkrrrppqpiqqpqrqrpresyqmghmrr

As of right now, we don't have a way in Python to confirm or identify which protein this sequence makes up. At the very least, this gives us an idea of what we're working with. We now define our function in Python that converts our sequence of letters to number:

```
def process_strings(c):
```

```
    '''Takes in a list of sequences 'c' and turns each one
       into a list of numbers.'''

    X = [] #create empty list to prepare list of numbers

    for  m, seq in enumerate(c):
        x = []
        for letter in seq:
            x.append(max(ord(letter)-97, 0)) #convert letter of
   amino acid sequence to a number, append to x

        X.append(x) #append amino acid sequence x to list of
   sequences X

    return X
```

Now we pass `X` and `Y` through this function:

```
#run our amino acid sequences through the process_strings function
X = process_strings(X)
Y = process_strings(Y)
```

We take a look at the first sequence in `X` to confirm the conversion:

```
print(X[0])
```

Output: `[12, 21, 11, 11, 17, ..., 7, 12, 17, 17]`
For the sake of brevity, output has been limited to the first and last five numbers. Cross referencing these numbers with the letters of the pre-converted sequence confirms the function did its job properly.

4 Analysis

We can plot a histogram of how the letters in the first antibody protein's amino acid sequence are distributed. Recall that we converted these letters to numbers, so these numbers are only representatives for these letters.

```
#view distribution of letters in the first antibody's amino acid
   sequence using 25 bins
plt.hist(X[0],25)
plt.xlabel("Number")
plt.ylabel("Frequency")
plt.show()
```

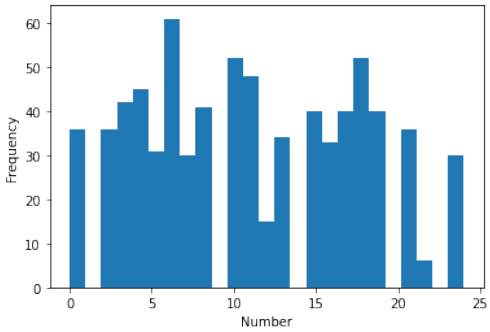

Figure 1: Distribution of letters in the first antibody protein's amino acid sequence

In the first antibody-related protein, the letters appear to be somewhat uniformly distributed, but there some obvious spikes and dips in other letters. Additionally, there are about 3 or 4 amino acids that are not contained in the entire sequence.

```
plt.hist(Y[0],25)
plt.xlabel("Number")
plt.ylabel("Frequency")
plt.show()
```

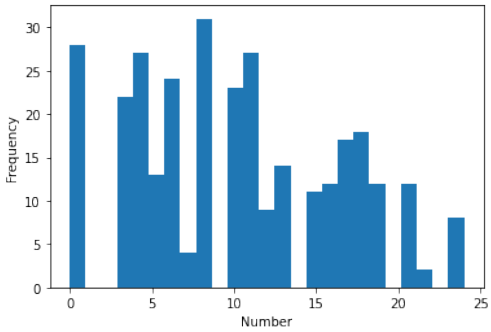

Figure 2: Distribution of letters in the first non-antibody protein's amino acid sequence

Like Figure 1, we use 25 bins again so that the frequency of each letter can be accounted for individually. The histograms in figures 1 and 2 appear quite similar to each other, with certain letters appearing more often than others, some only a few times, and others not at all.

We now extend this analysis to the length of every amino acid sequence in `X` and `Y`. We begin by creating lists containing the length for the amino sequences, one list each for `X` and `Y`.

```
# get a list of the length of each protein's amino acid sequence
X_lengths = [len(s) for s in X]
Y_lengths = [len(s) for s in Y]
```

We can take a closer look at these lists to find the longest amino acid sequences in our data:

```
np.max(X_lengths) #length of longest antibody protein sequence
```

Output: 5644

```
np.max(Y_lengths) #length of longest non-antibody protein sequence
```

Output: 11103

And the shortest sequences:

```
np.min(X_lengths) #length of shortest antibody protein sequence
```

Output: 5

```
np.min(Y_lengths) #length of shortest non-antibody protein sequence
```

Output: 6

To get a better idea of how these lengths are distributed across all of these sequences, we can plot histograms for `X_lengths` and `Y_lengths`.

```
plt.hist(X_lengths,bins=1000,range=(0,5000)); #distribution of
    lengths of our antibody amino acid sequences
plt.xlabel("Sequence Length")
plt.ylabel("Frequency")
plt.show()
```

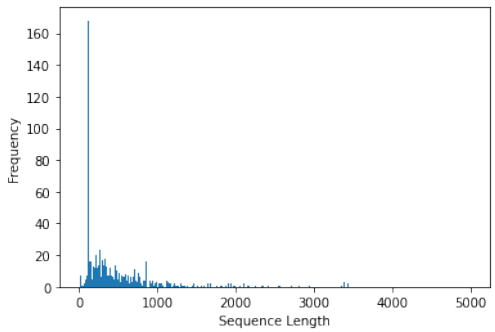

Figure 3: Distribution of lengths of antibody proteins' amino acid sequences

We can see that a large majority of proteins have a sequence of 1000 or less amino acids, with a very small amount of sequences exceeding a length of 1000.

Let's take a look at the distribution of the length of our non-antibody proteins:

```
plt.hist(Y_lengths,bins=1000,range=(0,5000)); #distribution of
    lengths of our non-antibody amino acid sequences
plt.xlabel("Sequence Length")
plt.ylabel("Frequency")
plt.show()
```

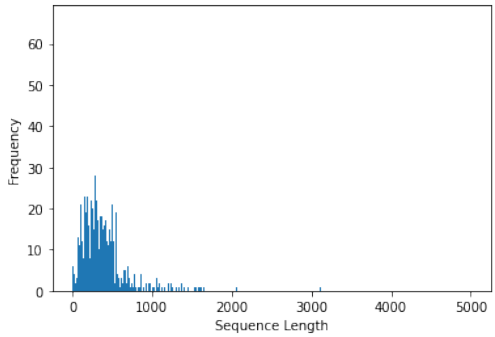

Figure 4: Distribution of lengths of non-antibody proteins' amino acid sequences

The histogram in 4 looks quite similar to the one in 3, but there are less proteins with a sequence length of over 1000 amino acids. Of the 2000 non-antibody proteins, it seems as though only 1 has a sequence length of over 3000, but the sequence of length 11,103 is not plotted.

Overall, the distribution of the length of proteins seems to be somewhat normally distributed with a noticeable right skew.

5 Conclusions

Using Python may prove to be quite useful in the analysis of proteins and their sequences. This may also assist us with determining a protein's higher order structures based on its sequence and amino acid distribution. Converting the letters to numbers proved to be quite useful in our analysis, as it provided us with the convenience of constructing histograms for both the individual sequences along with the lengths of these sequences.

References

[1] Goodsell, David, *The Machinery of Life*, (Copernicus Books, New York, 2009).