Lab 2: Turning Biology into Mathematics

Gil Cohen

September 7, 2020

1 Introduction

In this lab demonstration we will access the uniprot database and create a new data set. Our data set will consist of four thousand proteins, half associated with the keyword antibody and the other half not related to the keyword. The proteins are represented by their primary structure sequence of amino acids, in other words each protein is a string of letters representing each amino acid in the sequence.

The one-letter and three-letter codes for amino acids used in the knowledgebase are those adopted by the commission on Biochemical Nomenclature of the IUPAC-IUB

One-letter code Three-letter code Amino-acid name
A Ala Alanine
R Arg Arginine
N Asn Asparagine
D Asp Aspartic acid
C Cys Cysteine
Q Gln Glutamine
E Glu Glutamic acid
G Gly Glycine
H His Histidine
I Ile Isoleucine
L Leu Leucine
K Lys Lysine
M Met Methionine
F Phe Phenylalanine
P Pro Proline
S Ser Serine
T Thr Threonine
W Trp Tryptophan
Y Tyr Tyrosine
V Val Valine
O Pyl Pyrrolysine
U Sec Selenocysteine
B Asx Aspartic acid or Asparagine

Z Glx Glutamic acid or Glutamine
X Xaa Any amino acid

2 Procedures

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

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

#Getting the lengths of the lists X, Y
number_X = len(X)
number_Y = len(Y)

#This code prints the lengths of lists X and Y
print(number_X)
print(number_Y)

''' Amino acid sequence of the first protein in the list of proteins
    associated with 'antibody '''

X[0]


#Turn protein strings of letters into vectors of numbers:

def process_strings(c):
    ''' Takes in a list of sequences 'c' and turns each one
        into a list of numbers. '''

    X = [] #create empty list X

    for  m, seq in enumerate(c): #loop through each sequence seq in c
        x = [] #creating a local empty list x
        for letter in seq: #looping through each letter in sequence seq

            ''' appending a 0 for 'a', 1 for 'b'.. 25 for 'z' to x for each
                letter in seq '''

            x.append(max(ord(letter)-97, 0))

        #appending the sequence of numbers x to the list of lists X
```

```
            X.append(x)

        return X

# Turning X and Y into lists of number sequences
''' X contains the amino acid sequence of every protein associated with the
    word 'antibody', each represented by a sequence of numbers '''
''' Y contains the amino acid sequences of every protein unrelated to the
    word 'antibody', each represented by a sequence of numbers '''

X = process_strings(X)
Y = process_strings(Y)

''' This code prints the first protein's amino acid number sequence in the
    'antibody' group '''

print(X[0])

''' This code prints the first protein's amino acid number sequence in the
    non 'antibody' group '''

print(Y[0])
```

3 Analysis

```
''' This code plots a histogram showing the counts of 0, 1, .. 25
    representing a, b, .. z in the amino acid sequence of the first protein
    in the 'antibody' group '''

plt.hist(X[0],25)
```

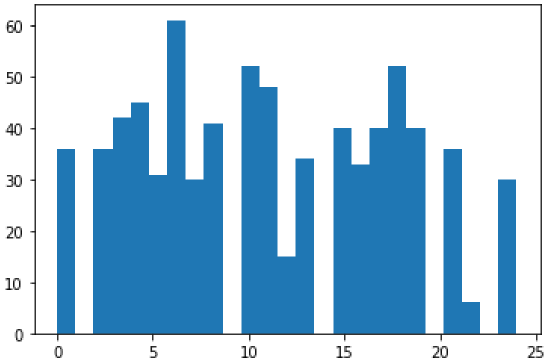

Figure 1: First 'antibody' protein amino acid counts

```
''' This code plots a histogram showing the counts of 0, 1, .. 25
    representing a, b, .. z in the amino acid sequence of the first protein
    in the non 'antibody' group '''

plt.hist(Y[0],25)
```

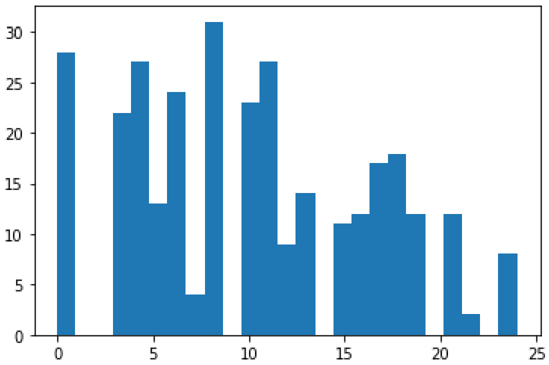

Figure 2: First non 'antibody' protein amino acid counts

```
''' This code computes the mean and standard deviation of the amino acid
    number sequence of the first protein in the 'antibody' group '''

np.mean(X[0]),np.std(X[0])

''' This is not meaningful, as the numbers are representing the letter
    codes of amino acids. Since the letter code for an amino acid is either
    a nominal or ordinal categorical variable, it is meaningless to find
    the mean and spread of the letter codes represented as numbers '''

#Converting the list X into an array and outputting its dimensions
np.array(X[0]).shape
-> (748,)

#Find lengths of all proteins:

X_lengths = [len(s) for s in X]
Y_lengths = [len(s) for s in Y]

#Finding the length of the longest protein in the 'antibody' group
np.max(X_lengths)
-> 5654

#Finding the length of the longest protein in the non 'antibody' group
```

```
np.max(Y_lengths)
-> 11103

#Finding the length of the shortest protein in the 'antibody' group
np.min(X_lengths)
-> 5

#Finding the length of the shortest protein in the non 'antibody' group
np.min(Y_lengths)
-> 6

''' This code plots a histogram showing the counts of all the lengths of
    proteins in the 'antibody' group '''

plt.hist(X_lengths, bins=1000,range=(0,5000));
```

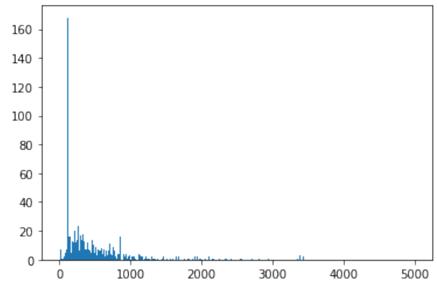

Figure 3: Protein length counts for all 'antibody' proteins

```
'''This code plots a histogram showing the counts of all the lengths of
   proteins in the non 'antibody' group '''

plt.hist(Y_lengths, bins=1000,range=(0,5000));
```

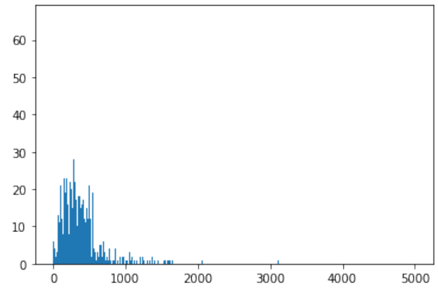

Figure 4: Protein length counts for all non 'antibody' proteins