Identifying amino acid predominant mutations in viruses by comparing matrices

A project submitted to the

Bioinformatics Centre

Savitribai Phule Pune University

For the degree of

M. Sc. in Bioinformatics

By

Ankit Mukherjee

Under the guidance of

Dr. Shekhar C. Mande

(Guide)

Distinguished Professor, Bioinformatics Centre

Month and year of submission
May 2024

CERTIFICATE

This is to certify that the project entitled **"Identifying amino acid predominant mutations in viruses by comparing matrices",** submitted by **Mr. Mukherjee Ankit** in partial fulfillment of the requirements for the degree of Master of Science in Bioinformatics, has been carried out satisfactorily by her/him at the Bioinformatics Centre, Savitribai Phule Pune University.

Date: 18/05/24
Place: Pune

Dr. Sangeeta Sawant
Director
Bioinformatics Centre
Savitribai Phule Pune University,
Pune 411007

CERTIFICATE

This is to certify that the project entitled **"Identifying amino acid predominant mutations in viruses by comparing matrices"**, submitted by **Mr. Mukherjee Ankit** in partial fulfillment of the requirements for the degree of Master of Science in Bioinformatics, has been carried out satisfactorily by her/him at the Bioinformatics Centre, Savitribai Phule Pune University, under my/our guidance and supervision.

Shekhar C. Mande

(Guide)
Distinguished Professor
Bioinformatics Centre

Date:
18/05/24
Place: Pune

DECLARATION & UNDERTAKING

I hereby declare that the project entitled, "**Identifying amino acid predominant mutations in viruses by comparing matrices**", submitted in partial fulfillment of the requirements of the degree of Master of Science in Bioinformatics, has been carried out by me at Bioinformatics Centre, Savitribai Phule Pune University under the guidance of **Shekhar C. Mande** (Guide). I further declare that the project work or any part thereof has not been previously submitted for any degree or diploma of any University.

I also declare that to the best of my ability, I have ensured that the submission made herein, including the main text, supplementary data, deposited data, database entries, software code, figures, does not contain any plagiarized material, content or ideas, and that all necessary attributions have been appropriately made and all copyright permissions obtained, cited and acknowledged.

I also declare that any further extension, continuation, publication, patenting or any other use of this project (either in full or in part), if any, shall be undertaken with prior written consent from the Director, Bioinformatics Centre, Savitribai Phule Pune University and the Project Supervisor/s.

I further state that I shall explicitly mention, "Bioinformatics Centre, Savitribai Phule Pune University" as "place of work" and acknowledge "the M.Sc. Bioinformatics training programme at Savitribai Phule Pune University for infrastructure and facilities" in the publication (print and online)/patent based on this work. I shall also acknowledge source of M.Sc. studentship (DBT or IGIB-GNR), if availed.

Date: 18/05/24 Ankit Mukherjee

Place: Pune BIM 2022_04

ACKNOWLEDGEMENTS

I extend my heartfelt appreciation to **Dr. Shekhar Mande** for generously dedicating his valuable time to contribute to my project. Additionally, I am immensely grateful to
Dr. Payel Ghosh for her unwavering support and guidance throughout the entire process.

Their expertise and assistance have been invaluable, and I am truly thankful to **Bithika Chatterjee**, my family and friends (**Pradnya**) for the success for my project.

Ankit Mukherjee
BIM_2022_04

TABLE OF CONTENTS

ABSTRACT

Viruses have a remarkable capacity to adapt new hosts and environments. This is highly dependent on their ability to generate spontaneous mutations in a short period of time. Identifying which mutations are significant that confer a selective advantage to the virus is essential not only for understanding the reason behind position specific frequent mutations but also serve as targets in vaccine development. To address this fact, we developed a novel approach to compare viral protein specific substitution matrices with standard matrices like Point Accepted Mutation (PAM) to identify statistically significant mutations. Our findings highlight that significant mutations often occur at the first and codon positions rather than the wobble position. Additionally, mutations frequently involve changes from amino acids having aliphatic side chains to those with aromatic side chains. These might be responsible for establishing new interactions between viral proteins and the host receptor, aiding in immune evasion or circumventing vaccine induced immunity. Monitoring these position specific mutations in viral proteins helps in understanding the evolution of viruses in lieu of the host-arms race and how this influences the host immune system, such as the evolution of the Human Leukocyte Antigen (HLA) locus through changes in immune receptors. Therefore, this method would be essential in rapid identification of significant mutations, thereby accelerating vaccine development.

LIST OF ABBREVIATIONS

- PAM: Point Accepted Mutation
- HLA: Human Leukocyte Antigen
- SARS-CoV-2: Severe Acute Respiratory Syndrome Related Coronavirus 2
- HCV: Hepatitis C Virus
- HBV: Hepatitis B Virus
- HIV: Human Immunodeficiency Virus
- dN/dS: Ratio of non-synonymous to synonymous substitution
- NCBI: National Centre for Biotechnology Information
- GISAID: Global Initiative on Sharing All Influenza Data
- CD-HIT: Cluster Database at High Identity with Tolerance
- MSA: Multiple Sequence Alignment
- MAFFT: Multiple Alignment using Fast Fourier Transform
- SD: Standard Deviation
- PDB: Protein Data Bank
- NTD: N-Terminal Domain
- RBD: Receptor Binding Domain
- MEGA: Molecular Evolutionary Genetics Analysis
- ACE2: Angiotensin Converting Enzyme 2

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

The rapid and continuous evolution of viruses presents formidable challenges to global public health. Viral mutations enable these pathogens to adapt swiftly to new hosts and environmental conditions, often leading to increased infectivity, enhanced transmissibility, and resistance to existing immune responses. Understanding the specific mutations that confer these advantages is crucial for developing effective vaccines and antiviral therapies. The mutation of an organism is defined as the probability that a change in genetic information is passed to the next generation. In viruses, a generation is often defined as a cell infection cycle, which includes attachment to the cell surface, entry, gene expression, replication, encapsidation, and release of infectious particles (*Sanjuán R & Domingo-Calap, 2016*). Rates of spontaneous mutation vary amply among viruses. Therefore, we mainly selected RNA viruses for study due to their high mutation rate, single-stranded genome, high availability of sequences and representative virus of different families (see *Table I*).

Table I: Viruses, their representative families and genome characteristics

Virus Family	**Representative Virus**	**Genetic Material**	**Strand**
Coronaviridae	*Coronavirus (SARS-CoV-2)*	RNA	Single Stranded
Flaviviridae	*Hepatitis C Virus (HCV)*	RNA	Single Stranded
Hepadnaviridae	*Hepatitis B Virus (HBV)*	DNA	Double Stranded
Orthomyxoviridae	*Influenza A Virus*	RNA	Single Stranded
Retroviridae	*Human Immunodeficiency Virus (HIV)*	RNA	Single Stranded

To classify proteins having a significant effect of these spontaneous mutations in the evolution of the virus, the ratio of non-synonymous to synonymous substitutions (dN/dS) served as a useful measure. It indicated the strength and mode of natural selection acting on

protein-coding genes, with $\omega > 1$ indicating positive (adaptive or diversifying) selection, $\omega = 1$ indicating neutral evolution, and $\omega \approx 1$ indicating negative (purifying) selection (*Jeffares DC et al., 2015*). The protein under selection pressure ($\omega > 1$) and control protein ($\omega \approx 1$) from the same virus were chosen for comparative analysis (see *Table II*).

Table II: Viruses and their corresponding proteins

Virus	**Protein under Selection Pressure**	**Protein presumably not under Selection Pressure**
Coronavirus (SARS-CoV-2)	Spike Protein	M or Matrix/Membrane Protein
Hepatitis C Virus (HCV)	E2 Protein	Core Protein
Hepatitis B Virus (HBV)	L Protein	Core Protein
Influenza A Virus	HA and NA Proteins	M1 Protein
Human Immunodeficiency Virus (HIV)	Env Protein	Gag Protein

The evolutionary process takes into consideration the frequency of change of each amino acid to another and the propensity of each to remain unchanged. This includes $20 \times 20 = 400$ possible comparisons. Point Accepted Mutation (PAM) matrix constructed from standard proteins documents 1572 changes in 71 groups of related proteins where each value are log-likelihood scores that reflect how likely one amino acid is substituted over the other (*Dayhoff MO et. al., 1978*). It is viewed as a Markovian model where a series of changes of state in a system are such that a change from one state to another does not depend on the previous history of the state. Following the same algorithm as PAM, matrices are formed from processed sequences of the proteins that are under selection pressure and control proteins from the same virus. It can be hypothesized that the matrix derived from protein under selection pressure would be significantly different from PAM matrix while the matrix derived from control proteins would be significantly similar. High-variability regions in protein sequences are mapped onto the corresponding viral proteins to identify sequence-level mutations that affect protein structure and function.

The significance of this research lies in several key areas. First, identifying these mutations enhances our understanding of the mechanisms by which viruses adapt and evade immune

responses. Such knowledge is critical for anticipating viral behavior and potential outbreaks. Second, the findings can directly inform the design and development of vaccines. By targeting mutations that confer significant advantages to the virus, vaccines can be made more effective against current and emerging strains. Third, this study provides insights into the co-evolutionary dynamics between viruses and their hosts, particularly how viral mutations impact host immune responses and receptor interactions.

MATERIALS & METHODS

Sequence Downloading and Filtering

Sequences of virus were downloaded from NCBI virus (https://www.ncbi.nlm.nih.gov/labs/virus/) and GISAID database (https://gisaid.org/) for Severe Acute Respiratory Syndrome 2 (SARS-CoV-2) virus with filters such as Complete, High Coverage, With Patient Status and Collection Data Complete.

Filters applied include:

- Using CD-HIT (command line tool, *Fu L et. al., 2012*) to keep only unique sequences by removing sequences which are 100 % similar.

  Command used: cd-hit -i input.fasta -o output.fasta -c 1.00 -n 5 -M 0 -T 0
- Trimming long headers

  Code (New Development):

```
import re


input_file = open('input.fasta','r')

a = open('output.fasta','w')

for line in input_file:

   if line.startswith(">"):

      header = re.search("\d+.\d+.\w+.\d+", line).group()

      a.write(">" + header + "\n")

   else:

      a.write(line)


input_file.close()

a.close()
```

- Filtering sequences by length (decided on the basis of the length of the protein, see *Table III*)

Table III: Viruses and length of their protein sequences

Virus	**Protein**	**Sequence Length (aa)**	
		Lower Limit	Upper Limit
Coronavirus (SARS-CoV-2)	Spike Protein	1200	1300
	M or Matrix/Membrane Protein	222	222
Hepatitis C Virus (HCV)	E2 Protein	300	400
	Core Protein	170	200
Hepatitis B Virus (HBV)	L Protein	300	400
	Core Protein	170	215
Influenza A Virus	HA Protein	550	570
	NA Protein	460	480
	M1 Protein	252	252
Human Immunodeficiency Virus (HV)	Env Protein	800	900
	Gag Protein	450	550

Code (New Development):

```
seq = ''


with open("input.fasta", "r") as input_file, \
  open("output_len_range.fasta", "w") as a, \
  open("output_seq_length.txt", "w") as b:
     for line in input_file:
        if line.startswith(">"):
           if seq != '':
              length = len(seq)
              b.write(header + seq + "\n" + "length = " + str(length) + "\n")
              if lowerlimit <= length <= upperlimit:
```

```
                a.write(header + seq + "\n")
            seq = ''
            header = line
        else:
            seq += line.rstrip()

    if seq != '':
        length = len(seq)
        b.write(header + seq + "\n" + "length = " + str(length) + "\n")
        if lowerlimit <= length <= upperlimit:
            a.write(header + seq + "\n")

a.close()
b.close()
```

- Removing sequences containing ambiguous characters

  Code (New Development):

```
input_file = open ('input.fasta', 'r')

a = open ('output_pure_seq.fas', 'w') # output file name containing umambiguous sequences

b = open ('output_ambi_seq.txt', 'w') # output file name containing ambiguous sequence headers

my_list = []

for line in input_file:
```

```
if line.startswith(">"):
    header = line
    seq = input_file.readline().rstrip()

    seq = seq[:len(seq) - 1]
    length = len(seq)
    c = 0
    for i in range(length):
        if seq[i] not in 'ARNDCEQGHILKMFPSTWYVarndceqghilkmfpstwyv-':
            c = c + 1
            my_list.append(seq[i])
    if c == 0:
        a.write(header + str(seq) +'\n')
    else:
        b.write(header + str(c) + '\n' + str(my_list) + '\n')
```

- Performing Multiple Sequence Alignment (MSA) using MAFFT (Command line tool, *Alqahtani, A. & Almutairy, M., 2023*)
  Command: mafft input.fasta > output.fasta

- Sequences containing large gaps were removed and MSA was performed again using MAFFT

  Code (New Development):

```
with open('input.fas', 'r') as f1, open('msa.fasta', 'r') as f2,
open('out_gap_removed.fasta', 'w') as output:

    seq = ''

    seq_ori = ''

    sequences = []

    sequences_ori = []

    header = []

    gap_count = 0

    list_gaps = []


    for line1 in f2:

        if line1.startswith(">"):

            if seq != '':

                sequences.append(seq)

                seq = ''

            header.append(line1.rstrip())

        else:

            seq += line1.rstrip()


    for line2 in f1:

        if line2.startswith(">"):

            if seq_ori != '':

                sequences_ori.append(seq_ori)

                seq_ori = ''

        else:

            seq_ori += line2.rstrip()
```

```
sequences.append(seq)
    sequences_ori.append(seq_ori)


    for sequence in sequences:
        for ntd in range(len(sequence)):
            if sequence[ntd] == '-':
                gap_count += 1
            else:
                if gap_count > 0:
                    list_gaps.append(gap_count)
                    gap_count = 0


        if any(gap >= 10 for gap in list_gaps):
            list_gaps = []
            continue
        else:
            position = sequences.index(sequence)
            output.write(header[position] + "\n" + sequences_ori[position] + "\n")
            list_gaps = []
```

Table IV: Virus Protein Sequences before and after filtering

Virus	**Protein**	**Number of Sequences**	
		Before Filtering	After Filtering
Coronavirus (SARS-CoV-2)	Spike Protein	14628247	22874
	M or Matrix/Membrane Protein	3822066	3723

Hepatitis C Virus (HCV)	E2 Protein	2635	1871
	Core Protein	5644	1366
Hepatitis B Virus (HBV)	L Protein	829	264
	Core Protein	21015	4283
Influenza A Virus	HA Protein	1141	474
	NA Protein	135125	40150
	M1 Protein	119100	5255
Human Immunodeficiency Virus (HIV)	Env Protein	328676	71783
	Gag Protein	204751	23351

Algorithm of PAM Matrix

- Filling up the accepted point mutation matrix with amino acid differences between each pair of sequences using MSA.

  Code (New Development):

```
filename = "input_msa.fasta"

alignment = AlignIO.read(filename, "fasta")

print(alignment)
```

- Assumption:

  Likelihood of amino acid X replacing Y is the same as that of Y replacing X. This is met by transposing the matrix and then dividing each element by 2. This makes the matrix symmetric.

  Code (New Development):

```
substitution_counts = {}

for i in range(alignment.get_alignment_length()):

    col = alignment[:, i]

    for j in range(len(col)):
```

```
        for k in range(j+1, len(col)):
            substitution_pair = (col[j], col[k]) # Creates a tuple
            substitution_counts[substitution_pair] =
substitution_counts.get(substitution_pair, 0) + 1


print(substitution_counts)
col_labels = ['A', 'R', 'N', 'D', 'C', 'Q', 'E', 'G', 'H', 'I', 'L', 'K', 'M', 'F', 'P', 'S', 'T', 'W', 'Y',
'V']
substitution_matrix = pd.DataFrame(0, index = col_labels, columns = col_labels)
for (aa1, aa2), count in substitution_counts.items():
   substitution_matrix.loc[aa1, aa2] = count


substitution_matrix.to_excel('substitution_matrix_selc_press.xlsx')
accepted_pmm = pd.read_excel("substitution_matrix_selc_press.xlsx", index_col =
0)


accepted_pmm += accepted_pmm.transpose()
accepted_pmm /= 2
```

- Calculating Relative Mutabilities of the amino acids:

  A complete picture of the mutational process must include a consideration of the amino acid that did not change as well as those that did. For this we need to know the probability that each amino acid will change in a given small evolutionary interval.

$$m(j) = \frac{\Sigma_{i=1, i \neq j}^{20} A(i,j)}{n(j)}$$

where,

m(j) = mutability of the jth amino acid

n(j) = frequency of the jth amino acid

A(i,j) = number of accepted point mutations from i to j or vice versa

Code (New Development):

```
A = 100 / ((accepted_pmm.loc[:,'A'].sum() - accepted_pmm.loc['A','A']) /
total_counter['A'])

col_labels.remove('A')



# Relative Mutabilities

relative_mutabilities = {}

relative_mutabilities['A'] = 100

for i in col_labels:

   relative_mutabilities[i] = round(((accepted_pmm.loc[:,i].sum() -
accepted_pmm.loc[i, i]) / total_counter[i]) * A)



relative_mutabilities
```

- Calculating normalized frequencies of all amino acids:

It indicates the relative frequency of exposure to mutation of the amino acid.

Background probability of each amino acid:

$$P(j) = \frac{\Sigma_{i=1, i \neq j}^{20} A(i_J j)}{m(j)}$$

Normalization is carried out by dividing each value by the sum of the background frequencies:

$$f(j) = \frac{p(j)}{\sum p(j)}$$

Code (New Development):

```
# Calculating frequencies of all amino acids

freq = {}

for j in col_labels:

   freq[j] = (accepted_pmm.loc[:,j].sum() - accepted_pmm.loc[j,j]) /
relative_mutabilities[j] # or dayhoff_relative_mutabilities



sum_freq = sum(freq.values())

for i, j in freq.items():

   freq[i] = round((j / sum_freq), 3)



freq
```

- Construction of the Mutation Probability Matrix:

  An element of this matrix gives the probability that the amino acid in column j will be replaced by amino acid i after a given evolutionary interval.

  Choice of constant of proportionality λ:

  Parameter to control how much change is allowed during analysis. To find the mutation probability matrix for PAM_1 matrix, the requirement that 99 % of the

amino acid in a sequence are conserved is imposed. The quantity n(j)M(j,j) is equal to the number of conserved amino acid j units:

$$\sum_{j=1}^{20} n(j)M(j,j) = \sum_{j=1}^{20} n(j) - \lambda \sum_{j=1}^{20} n(j)m(j) = N - N\lambda \sum_{j=1}^{20} f(j)m(j)$$

$$0.99 = 1 - \lambda \sum_{j=1}^{20} f(j)m(j)$$

Code (New Development):

```
# Calculating choice of proportionality lambda

s = 0

for i in col_labels:

   s += freq[i] * relative_mutabilities[i] # or dayhoff_relative_mutabilities



prop_lambda = (1 - 0.99) / s

prop_lambda
```

Non - Diagonal Elements:

$$M(i,j) = \lambda A(i,j) \frac{m(j)}{\sum_{i=1, i \neq j}^{20} A(i,j)}$$

Code (New Development):

```
# Creating the mutation matrix

mutation_matrix = pd.DataFrame()

for i in col_labels:

    for j in col_labels:

        mutation_matrix.loc[i,j] = (prop_lambda * relative_mutabilities[j] *
accepted_pmm.loc[i,j]) / (accepted_pmm.loc[:,j].sum() - accepted_pmm.loc[j, j])



mutation_matrix
```

Diagonal Elements:

$$M(j,j) = 1 - \sum_{i=1, i \neq j}^{20} M(i,j)$$

simplifies to,

$$M(j,j) = 1 - \lambda m(j)$$

Code (New Development):

```
for i in col_labels:

    for j in col_labels:

        if i == j:

            mutation_matrix.loc[i,j] = 1 - (prop_lambda * relative_mutabilities[j])



mutation_matrix
```

- Construction of PAM_n matrix:

$$M_n = M_1^n$$

Code (New Development):

```
pam_n = mutation_matrix.copy()

pam_1 = mutation_matrix.copy()



for _ in range(250):

   pam_n = pam_n.dot(pam_1)



pam_n
```

- Construction of log odds matrix:

A log odds positive value indicates true accepted point mutation while a log odds negative value indicates mutation due to random chance.

$$\mathrm{PAM}_n(i,j) = log\frac{f(j)M_n(i,j)}{f(i)f(j)} = log\frac{f(j)M^n(i,j)}{f(i)f(j)} = log\frac{M^n(i,j)}{f(i)}$$

Code (New Development):

```
for i in col_labels:

   for j in col_labels:

      pam_n.loc[i, j] = round((math.log10(pam_n.loc[i, j] / freq[i])) * 10)



pam_n.to_excel('pam_250_selc_press.xlsx')
```

Protein Sequence Similarity

Dayhoff PAM matrix derived the Accepted Point Mutation matrix using sequences that were at least 85 % identical. Protein Sequence Similarity was assessed using "Align two or more sequences" option in blastp.

Calculating d_n/d_s ratio

Proteins chosen for study were based on the ratio of non-synonymous to synonymous mutations based on literature survey. Proteins which showed $\omega > 1$ were considered to be proteins under positive selection pressure and prone to frequent mutations accepted through natural selection while proteins which showed $\omega \approx 1$ were considered to be control proteins as they were under neutral selection pressure. However, to assess whether they truly represent the same, MEGA software was used with nucleotide sequences as input.

Selection analysis of aligned sequences using Z-test of selection with filters such as:

- Scope: Overall Average
- Test Hypothesis (HA: Alternative): Positive Selection (HA: dN > dS) or Test of Neutrality (HA: dN ≠ dS)
- Bootstrapping using 100 replications
- Model: Modified Nei – Gojobori method (Proportion)
- Gaps / Missing Data Treatment: Pairwise Deletion

The analysis gives an output of probability. It must be < 0.05 for hypothesis rejection at 5 % level.

Identifying Significant Mutations

Chi-squared test was carried out to assess whether the matrices compared were significant or not.

Code (New Development):

```
# Significance test - chi-squared test between prot_selc_press(observed) and
prot_dayhoff(expected) at p = 0.05

# Upper Triangular Matrix

chi_squared = 0

size = len(col_labels)

for i in col_labels:

   for j in range(col_labels.index(i), size):

      chi_squared += ((prot_selc_press[col_labels[j]][i] - prot_dayhoff[col_labels[j]][i]) **
2)/prot_dayhoff[col_labels[j]][i]



print(f"The chi-squared value at 95 % significance level is: {chi_squared}")

# The matrix of prot_selc_press is significantly different from the dayhoff pam250 matrix

# The chi-squared value is much higher than the value at dof = 19 and p = 0.05, 30.144
```

Calculating difference matrix and Standard Deviation (SD) to assess which mutations show values higher than the value in difference matrix:

Code (New Development):

Calculating average differences:

```
# Average differences between prot_selc_press and prot_dayhoff

sum_diff = 0

for i in col_labels:

   for j in col_labels:

      sum_diff += (prot_selc_press[i][j] - prot_dayhoff[i][j])



avg_sum_diff = sum_diff / 400

avg_sum_diff
```

Calculating SD:

```
# Standard Deviation of prot_selc_press

var = 0

for i in col_labels:

    for j in col_labels:

        var += ((prot_selc_press[i][j] - avg_sum_diff) ** 2) / 400


sd = math.sqrt(var)

sd
```

Difference Matrix:

```
diff_matrix_selc_dayhoff = pd.DataFrame()

for i in col_labels:

    for j in col_labels:

        diff_matrix_selc_dayhoff.loc[i, j] = prot_selc_press[i][j] - prot_dayhoff[i][j]


diff_matrix_selc_dayhoff
```

Identifying values more than SD in the difference matrix:

```
for i in col_labels:

    for j in col_labels:

        if diff_matrix_selc_dayhoff[i][j] > sd:

            print(diff_matrix_selc_dayhoff[i][j], i, j)
```

Calculating Entropy at every sequence position

Entropy or Shannon Entropy in sequence analysis refers to the measure of the variation of characters (column) in multiple sequences. The more conserved a column, the smaller is its entropy.

$$H = -\sum p(x)\log p(x)$$

where, H = Shannon entropy, p(x) = Frequency of every amino acid at a given position

Code (New Development):

```
entropy = []


for i in range(alignment.get_alignment_length()):
    entropy_cal = 0
    freq = Counter()
    col = alignment[:, i]
    freq.update(col)
    for i in freq.values():
        entropy_cal += -(i * math.log2(i))
    entropy.append(entropy_cal)
plt.xlim(1, alignment.get_alignment_length())
plt.plot(entropy)
plt.xlabel("Sequence Position")
plt.ylabel("Entropy")
plt.title("Entropy for every position")
plt.show()
```

Filtering variable positions i.e. positions with high entropy values with an assumed cutoff

Code (New Development):

```
print("The variable positions are:")
for i in entropy:
   if i > cutoff:
      print(entropy.index(i))
```

Calculating Codon Usage:

Codon usage bias refers to the differences in the frequency of occurrence of synonymous codons in a coding DNA. Codon usage bias can preferentially use one synonymous codon among other synonymous codons of an amino acid (*Parvathy ST, 2022*). It was calculated for every protein sequence using Codon Usage (https://www.bioinformatics.org/sms2/codon_usage.html) server with nucleotide sequence as input.

Mantel Test

The Mantel Test is a statistical test that measures the correlation between two matrices. The basic procedure involves converting each data matrix to a dissimilarity matrix and calculate the correlation between the two dissimilarity matrices (correlation ranges between -1 and +1). It is implemented using *vegan* package in R.

Code (New Development):

```
# Mantel test in R

library(vegan)

library(readxl)


prot_selc_press <- read_excel("pam_250_selc_press_wr.xlsx")

control <- read_excel("pam_250_control_wr.xlsx")

pam <- read_excel("pam_250_wr.xlsx")


dist1 <- dist(prot_selc_press, method = "euclidean")

head(dist1)


dist2 <- dist(control, method = "euclidean")

head(dist2)


dist3 <- dist(pam, method = "euclidean")

head(dist3)


comp1 <- mantel(dist1, dist3, method = "spearman", permutations = 9999, na.rm = TRUE)

comp1


comp2 <- mantel(dist2, dist3, method = "spearman", permutations = 9999, na.rm = TRUE)

comp2
```

RESULTS & DISCUSSION

Analysis of Significant Mutations

Significant mutations obtained by comparing matrix derived from protein under selection pressure and PAM250 were for example, Cysteine to Phenylalanine, Phenylalanine to Tryptophan, Alanine (Glutamic Acid, Isoleucine, Methionine, Glycine) to Tryptophan.

Cysteine has two codons TGT and TGC among which TGC is a preferential codon in HBV while Phenylalanine also has two codons TTC and TTT among which TTC is a preferential codon. The mutation involves a 2nd base change. Similarly, Methionine has one codon ATG mutating to Tryptophan codon TGG which involves both 1st as well as 2nd base change.

In hemagglutinin protein of Influenza A virus, Alanine having four codons (GCG, GCA, GCT and GCC) among which GCA and GCT are preferential mutating to Tryptophan codon TGG involves change in all three positions. Moreover, this is a change of an amino acid bearing an aliphatic side chain to an amino acid having aromatic side chain which might disrupt several interactions and form new ones. Similarly, is the change from Isoleucine to Tryptophan.

On the other hand, when comparing matrix derived from control and PAM250, we get significant mutations as Cysteine to Tryptophan (Leucine, Methionine). Cysteine has two codons TGT and TGC among which TGT is preferential in HBV. Therefore, TGT mutating into TGG (Tryptophan) is a change in the 3rd base position i.e. wobble base. We know change in the wobble base is least affected and this defines the degeneracy of genetic code.

Cysteine mutating into Leucine also involves change in 2nd and 3rd base position but as their side chains are both aliphatic in nature, the mutations might not be significant. Moreover, this mutation might occur when the sulfhydryl group in cysteine is not playing a major role.

Cysteine mutating into Methionine won't affect the structure of the protein because both have aliphatic side chains and a sulfhydryl group due to which no interactions would be affected.

Protein Sequence Similarity

Using blastp it was verified that all sequences considered for analysis were at least 85 % identical. This helped in applying PAM algorithm to the filtered sequences for deriving the matrices.

Protein Under Selection Pressure and Control Protein

The fact that proteins under selection pressure must have $\omega > 1$ and control protein $\omega \approx 1$ was satisfied by using the modified Nei – Gojobori approach (proportion) and thus, the sequences can be considered for further analysis.

Variability in protein sequence

The position with high entropy values is more variable and more prone to changes in the DNA sequence which ultimately gets reflected in the protein structure and thus the function of the protein.

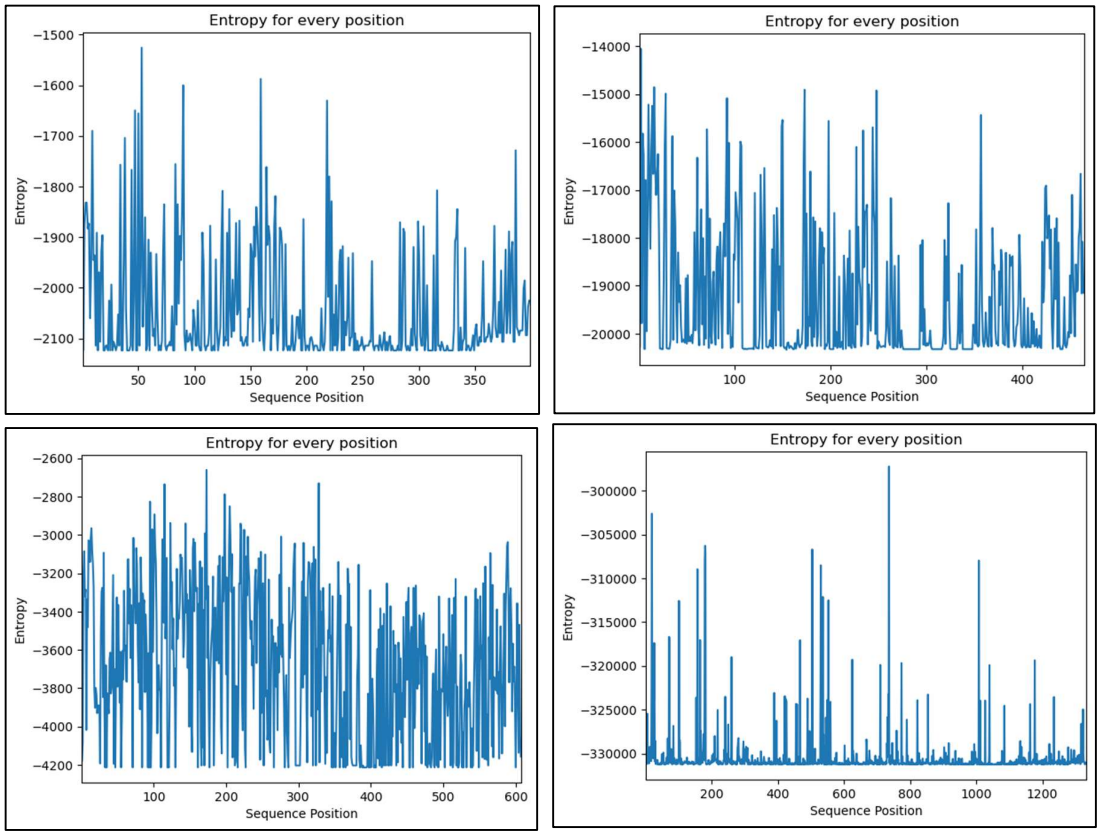

Figure 1: Entropy for every position of (Top left) HBV protein preS, (Top right) HCV protein E2, (Bottom left) Influenza HA, (Bottom right) SARS-CoV-2 Spike Protein

The most variable positions were mapped onto the respective proteins using Pymol software. These proteins were curated using blastp of virus protein sequences against PDB database with the best alignment scores and E value.

Table V: Virus proteins and their PDB IDs

Virus	**Protein**	**PDB ID**
Coronavirus (SARS-CoV-2)	Spike Protein	7KRQ
Hepatitis B Virus (HBV)	L Protein	7TUK
IInfluenza A Virus	HA Protein	6IDD
	NA Protein	6IXK

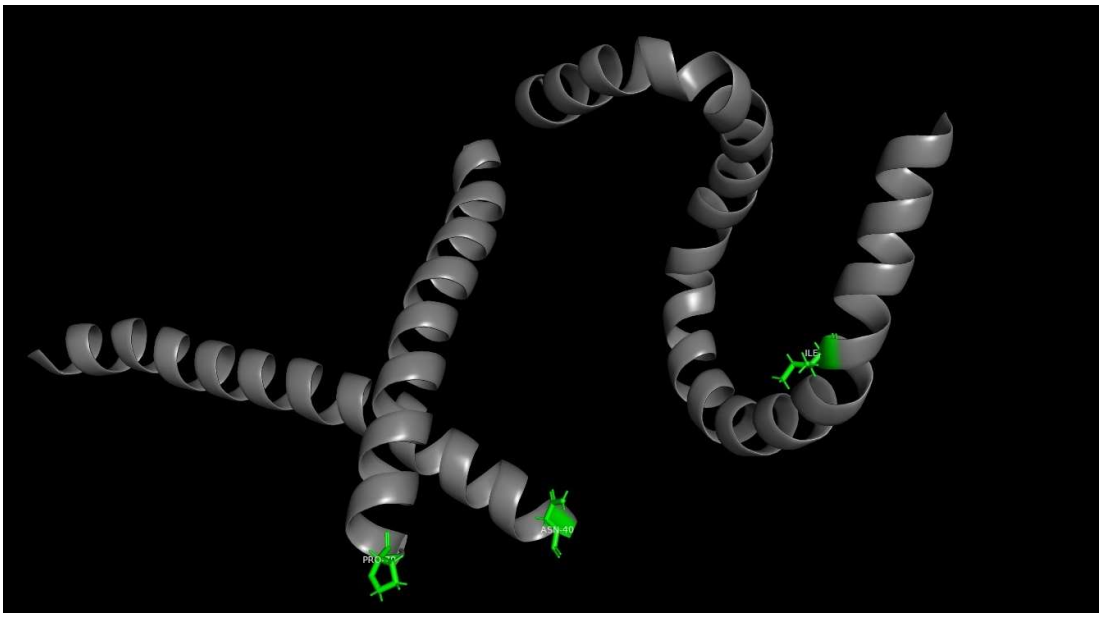

Figure 2: Mutations labelled and highlighted as green with stick representation for HBV protein preS

The mutations in HBV protein are mainly located in α-helix region which are secondary structures of the protein that might play an essential role in interactions with the host receptor.

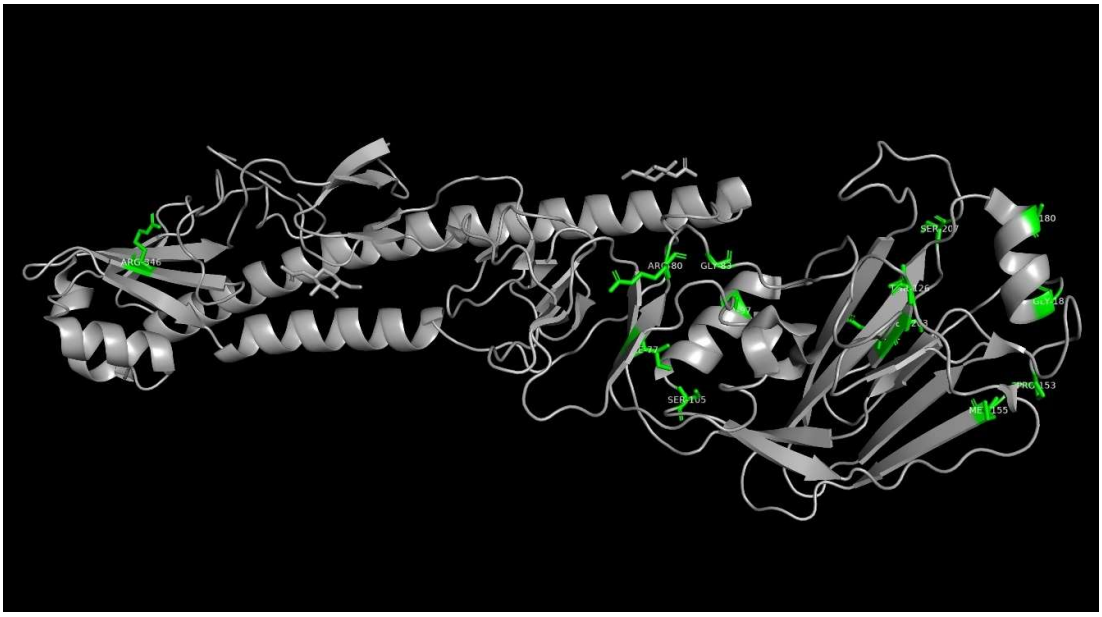

Figure 3: Mutations labelled and highlighted as green with stick representation for Influenza protein Hemagglutinin

In Hemagglutinin, the mutations are dispersed among the secondary structures α-helix, β-sheets as well as the coiled coil regions.

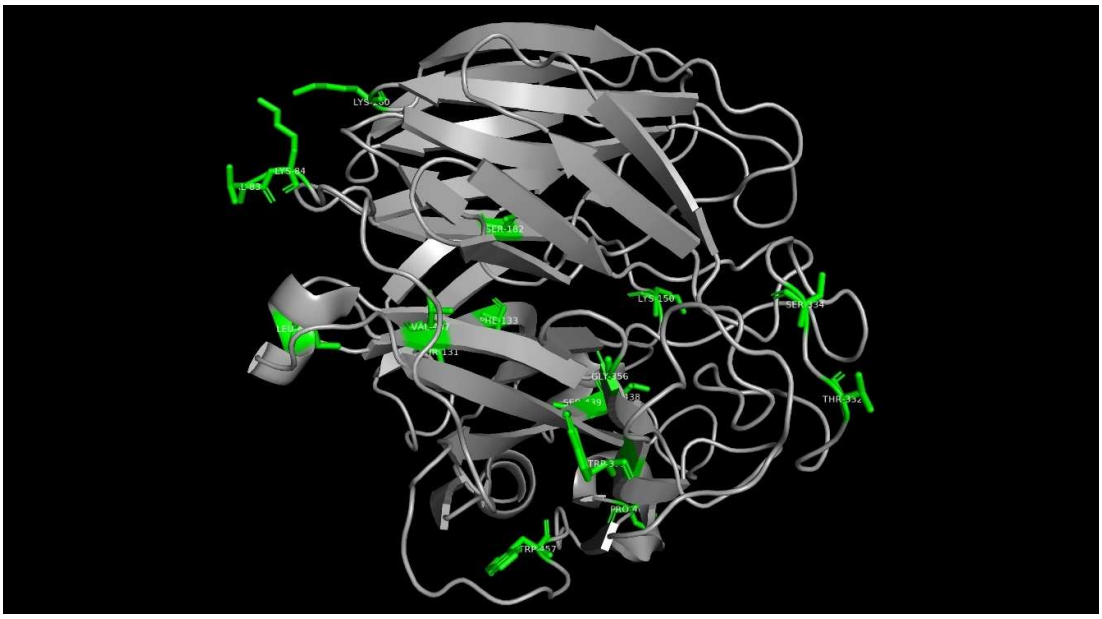

Figure 4: Mutations labelled and highlighted as green with stick representation for Influenza protein Neuraminidase

Most of the mutations in neuraminidase are constricted in the loop regions which are variable and are prone to frequent mutations.

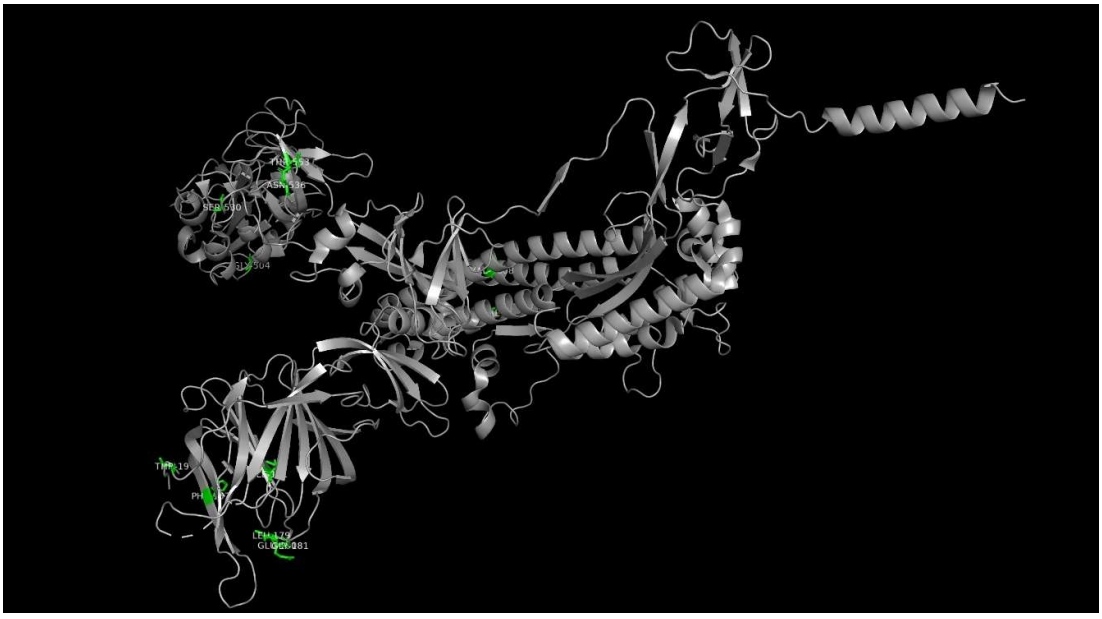

Figure 5: Mutations labelled and highlighted as green with stick representation for SARS-CoV-2 Spike protein

Mutations in spike protein are mostly in the Receptor Binding Domain (RBD) and N-terminal domain (NTD) which plays an important role in virus infection. NTD is responsible for virus attachment and are valuable targets for neutralizing antibodies. Similarly, RBD allows coronaviruses to bind to target body receptors (such as ACE2 on respiratory epithelial cells) and enter cells to cause infection.

Mantel Test

Although the test should have shown that matrices derived from proteins under selection pressure have lower correlation with PAM250 compared to matrices derived from control proteins, the results did not support this.

CONCLUSIONS

- Significance of Specific Mutations: The study identifies significant mutations in viral proteins that confer selective advantages to viruses. These mutations are critical for understanding viral adaptation to new hosts and environments. The novel approach used compares viral protein-specific substitution matrices with standard matrices like PAM (Point Accepted Mutation), revealing statistically significant mutations.

- Mutation Characteristics: Significant mutations are often found at the first and second codon positions rather than the wobble (third) position. These mutations frequently involve changes from amino acids with aliphatic side chains to those with aromatic side chains. Such changes may enhance interactions between viral proteins and host receptors, aiding in immune evasion and overcoming vaccine-induced immunity.

- Impact on Viral Evolution and Host Immune Response: Monitoring these position-specific mutations helps in understanding the co-evolution of viruses and their hosts. The study emphasizes the influence of viral mutations on the evolution of host immune systems, such as the Human Leukocyte Antigen (HLA) locus, and how these mutations affect immune receptor interactions.

- Application in Vaccine Development: The method developed in this study is essential for the rapid identification of significant mutations, which can accelerate the process of vaccine development. By pinpointing mutations that confer advantages to the virus, researchers can target these mutations in vaccine design, potentially improving vaccine efficacy.

FUTURE WORK

- Expansion to Other Viruses: Extend the methodology to analyze mutations in a wider range of viruses, including those with different genomic structures and replication mechanisms. This can help validate the approach across diverse viral families and provide a broader understanding of viral evolution.
- Refinement of Substitution Matrices: Further refine the substitution matrices by incorporating more extensive and diverse sequence data. This could improve the accuracy of identifying significant mutations and enhance the predictive power of the model.
- Integration with Structural Biology: Combine the findings with structural biology techniques to understand how identified mutations affect protein structure and function. This can provide insights into the molecular mechanisms by which mutations influence viral behavior and host interactions.
- Real-Time Monitoring: Develop tools for real-time monitoring of viral mutations using the proposed method. This can be particularly useful for emerging viral outbreaks, enabling rapid identification of significant mutations and informing public health responses.
- Application in Vaccine and Drug Design: Utilize the identified mutations as targets for vaccine and antiviral drug development. By focusing on mutations that confer selective advantages to viruses, new therapeutic strategies can be devised to counteract these adaptations.
- Host-Virus Interaction Studies: Investigate the impact of identified mutations on host-virus interactions in more detail. This includes studying how mutations affect immune evasion, receptor binding, and other aspects of viral pathogenicity.
- Evolutionary Studies: Conduct evolutionary studies to trace the origins and spread of significant mutations across different geographical regions and host species. This can help in understanding the evolutionary pressures driving these mutations.

- Data Integration and Machine Learning: Integrate the mutation data with other omics data (e.g., transcriptomics, proteomics) and apply machine learning techniques to uncover patterns and predict future mutation trends.

By pursuing these future directions, the methodology and findings from this study can be expanded and applied to a broader range of viral research, ultimately contributing to better preparedness and response strategies against viral diseases.

References

Articles:

- Alqahtani, A.; Almutairy, M. Evaluating the Performance of Multiple Sequence Alignment Programs with Application to Genotyping SARS-CoV-2 in the Saudi Population. Computation 2023, 11, 212. https://doi.org/10.3390/computation11110212
- Cuypers L, Li G, Libin P, Piampongsant S, Vandamme AM, Theys K. Genetic Diversity and Selective Pressure in Hepatitis C Virus Genotypes 1-6: Significance for Direct-Acting Antiviral Treatment and Drug Resistance. Viruses. 2015 Sep 16;7(9):5018-39. doi: 10.3390/v7092857. PMID: 26389941; PMCID: PMC4584301.
- Dayhoff MO, Schwartz RM, Orcutt BC (1978). "A model of Evolutionary Change in Proteins". Atlas of protein sequence and structure (volume 5, supplement 3 ed.). Washington, DC.: National Biomedical Research Foundation. pp. 345–358. ISBN 978-0-912466-07-1
- Emam M, Oweda M, Antunes A, El-Hadidi M. Positive selection as a key player for SARS-CoV-2 pathogenicity: Insights into ORF1ab, S and E genes. Virus Res. 2021 Sep;302:198472. doi: 10.1016/j.virusres.2021.198472. Epub 2021 Jun 10. PMID: 34118359; PMCID: PMC8190378.
- Fu L, Niu B, Zhu Z, Wu S, Li W. CD-HIT: accelerated for clustering the next-generation sequencing data. Bioinformatics. 2012 Dec 1;28(23):3150-2. doi: 10.1093/bioinformatics/bts565. Epub 2012 Oct 11. PMID: 23060610; PMCID: PMC3516142.
- Jeffares DC, Tomiczek B, Sojo V, dos Reis M. A beginners guide to estimating the non-synonymous to synonymous rate ratio of all protein-coding genes in a genome. Methods Mol Biol. 2015;1201:65-90. doi: 10.1007/978-1-4939-1438-8_4. PMID: 25388108.
- Li, S., Wang, Z., Li, Y. et al. Adaptive evolution of proteins in hepatitis B virus during divergence of genotypes. Sci Rep 7, 1990 (2017). https://doi.org/10.1038/s41598-017-02012-8
- Louie RHY, Kaczorowski KJ, Barton JP, Chakraborty AK, McKay MR. Fitness landscape of the human immunodeficiency virus envelope protein that is targeted

by antibodies. Proc Natl Acad Sci U S A. 2018 Jan 23;115(4):E564-E573. doi: 10.1073/pnas.1717765115. Epub 2018 Jan 8. PMID: 29311326; PMCID: PMC5789945.

- Parvathy ST, Udayasuriyan V, Bhadana V. Codon usage bias. Mol Biol Rep. 2022 Jan;49(1):539-565. doi: 10.1007/s11033-021-06749-4. Epub 2021 Nov 25. PMID: 34822069; PMCID: PMC8613526.
- Sanjuán R, Domingo-Calap P. Mechanisms of viral mutation. Cell Mol Life Sci. 2016 Dec;73(23):4433-4448. doi: 10.1007/s00018-016-2299-6. Epub 2016 Jul 8. PMID: 27392606; PMCID: PMC5075021.
- Tusche C, Steinbrück L, McHardy AC. Detecting patches of protein sites of influenza A viruses under positive selection. Mol Biol Evol. 2012 Aug;29(8):2063-71. doi: 10.1093/molbev/mss095. Epub 2012 Mar 16. PMID: 22427709; PMCID: PMC3408068.

URLs:

- https://www.ncbi.nlm.nih.gov/labs/virus/
- https://gisaid.org/
- https://blast.ncbi.nlm.nih.gov/Blast.cgi
- https://www.bioinformatics.org/sms2/codon_usage.html