

Re-implementing ExpCM Likelihood Function from PhyDms

Ian Chang¹, Shubham Mohole², Hannah Price³, Rishi Shah⁴, and Evan Cui⁵

¹Biometry & Statistics, 2027, ic429

²Computer Science, 2026, sam588

³Computer Science, 2026, hep43

⁴Computer Science, 2027, rrs244

⁵Computer Science, 2027, ec875

ABSTRACT

PhyDMS is a phylogenetic software that uses Experimentally Informed Codon Models (ExpCMs) for site-specific modeling of gene selection and evolution. This contrasts traditional codon models like YNGKP which assume similar constraints across a gene driving selection, ExpCMs integrate laboratory-measured Deep Mutational Scanning (DMS) results, which tracks the amino acid preferences for each site on the gene to better capture diverse selective pressures. In this project, we reimplement the ExpCM likelihood function and downstream stringency parameter optimization. We validate our implementation on HIV-1 Env (BF520 and BG505 strains) and Influenza A hemagglutinin datasets. These represent viral surface glycoproteins, which evolve rapidly under immune pressure and are thus a prime candidate to study gene selection and evolution. We reproduce published PhyDMS results in Haddox (2018) and Lee (2018), following the documentation provided by the Bloom Lab (2017) about the program's numerical implementation. We compare our re-implementation's log-likelihood values against original PhyDMS, to provide statistically strong evidence that our program extracts meaningful information from DMS data as compared with a null baseline. We additionally compare site-by-site likelihood contributions against the original software output to discover specific sites' differing selective pressures and analyze our re-implementation's success with particularly high entropy sites. Finally, we compare with the original PhyDMS' β or stringency parameter over an MLE process, assessing how our re-implementation is interpreting the degree of the experimental "signal". We found our re-implementation was correct through comparable global log likelihoods and ω/β parameters. On a site-wise scale, the site-wise likelihood contributions were also similar (with a slope of 0.75 to 0.9 indicating scaling with the original ExpCM). As a whole, this re-implementation successfully demonstrates the advantages of ExpCMs over standard codon models, particularly the ability to identify biologically meaningful sites and reproduce expected trends in stringency with nature. This biologically-rooted framework for quantifying how natural viral evolution and lab-measured mutational effects are related has major real world implications, including allowing researchers to predict high-probability viral mutations and previously unknown selective pressures.

Keywords (minumum 5): Phylogenetic Modeling, Deep Mutational Scanning, Experimentally Informed Codon Models, Felsenstein's pruning algorithm, HIV and Influenza

Project type: Re-implementation of ExpCM Likelihood Function

Project repository: <https://github.com/evcui/CS4775-Project>

AI Use Attribution Statements:

Shubham Mohole. I used Gemini (November 12, 2025) with the prompt: "I am running `dms_tools2.utils.renumberSites()` and i think after looking at my personal debug logs there might be issue with pandas function within in said function. I am looking at the utils file and seems a github issue with prior version of pandas on the append but not sure if that is already resolved as note says or i have to resolve it because i know append maybe deprecated. Should I just replace with `pd.concat` and do i just need to reassign function in the notebook with this fixed `renumberSites` function i write? Also there was some error with convert objects so is that also deprecated and seems like i dont need it." The AI explained to me the pandas package issues and the differences in append and concat and highlighted how the dmstools was using deprecated calls as well as with the pandas convert objects. It confirmed that my fix was correct and that I can just fix the function in my notebook cell. The implementation is still my work sourced from the dmsutils github.

Shubham Mohole. I used Gemini (November 13, 2025) with the prompt: "I am running RAXML-NG command and in

documentation they want CPUs how do i get that slurm cpus is there like a variable name i can dollar sign in or what to put -threads flag. Also what difference between RAXML and RAXML-NG and what is difference with other commonly used models in literature? I am reproducing some codon substitution models and the other people had used RAXML so confirm that this is okay also in documentation its say prefix so is that prefix for fasta file or I am confused a little bit or just naming so just any letter like T - does it want character or string? Do I also need to backslash if I create a new line when entering a parameter in a .sh file and also can you explain to me the relationship for filenames like the \$ like when to put inside the quotation marks, when to put outside and when if wanting to place in directory like have another slash, all these details regarding the slurm. Also when doing conda activate would i need to like first source the environment file i have or thats taken care of by this conda environment i have sestup but i dont wat it erroring on what conda is so what source command do I need to run and i am concered i have multiple enviornments going on so regardless of what directory i am in i want this specific haddox enviorenment withe conda to be activated? Also what flag do i use next mkdir so that it overwrites folders instead of erroring?i have these settings defined, there are all that are needed right, `#!/bin/bash #SBATCH --cpus-per-task=16 #SBATCH --mem=128G #SBATCH --time=12:00:00 #SBATCH --output=haddox_run.log`?" The AI said to use GTR+G because it is more precise than an approximation approach and said that this new version is better than the older versions implemented by the two papers. I consulted the RAXML source paper for further clarification. The AI said that the backslashes are important when doing multi-line argument filling and explained to me in detail the syntax for \$ in sh files as well as told me about the -p flag. For the CPU for the slurm it said to use the \$SLURM_CPUS_PER_TASK command. It said run `source ~/.bashrc` before activating the conda and then do `conda activate` and said my settings were sufficient. The slurm commands are still my own work.

Shubham Mohole. I used Gemini (November 19, 2025) with the prompt: "For the Sars2Cov dataset by the Bloom Lab I tried using this single_mut_effects.csv where we have the mutational effect data with columns:"site_RBD","site_SARS2","wildtype","mutant","mutation","mutation_RBD","bind_lib1","bind_lib2","bind_avg", "expr_lib1","expr_lib2","expr_avg". as a DMS dataset source. I modified it so that none of the preference effects are negative with adding by a constant and normalized the preference data so sum to 1 for each site. I also using the bind_avg column because I read that the ACE2 binding can be the proxy for a virus being effective. I added neutral preferences as well so that the numbers match up with gene sequence but at the end it just gave beta as 0 so I dont think this data is useful for my attmpts at reimplementing the phydms likelihood functionality. Can you confirm that such an interpretation of the beta as 0 is correct?" The AI agreed with assesment of the beta value and motivated me to just focus on other validation datasets.

Shubham Mohole. I used Gemini (December 10, 2025) with the prompt: "How do I make sure that for the python plots the labels on bottom do not get cutoff is there a figure resizing option? Also i want to make bar graph where for some labels on the x-axis it has two bars coming out of it instead of 1. Now my thinking right now i am just going to create a dataframe with two keys and one of the keys i will have a list of 0s corresponding to the values in the other key's list for which i just want one bar and then for the index i want the two bars i just have the value and then i will specify index of the plt to know labels, is that right appraoch or would the python plot not pick up on this sort of dataframe? Also should I always use numpy arrays for efficiency and to clarify i can use ' ' for even the indexing right because the columns are letters or doesnt matter? Also i want your feedback on what the best caption, titles should be for these figures the first figure is the hiv comparison of models on their log-likelihoods and similar for influenza but log-likelihood i feel like is there a standard accepted acronym for it? Also how do i make it clear that selection stringency and beta are same thing in figures should there be equal sign or should i say like selection stringency beta but then its not just beta since beta is parameter and the thing on the axis is values. Also for the another figure i plan on having it as phydms dLnl (dellta log liklihoollod) vs our dLNL but title as sitewise dLNL and then specific maybe in parentheses the strain type and then my last figure i plan on saying like sitewise model improvement or should i include expcm-null to make it clear?" The AI told me above various options such as defining figsize, using the tightlayout function as well as use cases for the numpy array. The AI gave me suggestions which I used for the x-axis, y-axis and title labels for each of the figures which I used for initial draft but then I then rewrote it myself. The AI told me that I should define my dataframe like this: `(pd.DataFrame(dictionary, index = []))` where in the dictionary I define the number of bars at each point on x-axis with the number of keys in dictionary which was aligned with what I was thinking of. I rewrote the code and the implementation is still my own work.

Shubham Mohole. I used Gemini (December 12, 2025) with the prompt: "So i would like to have a way to distinguish between an omega parameter that is a list versus that is just a value how do I do that if i know the list should be an np array that I am passing if I do pass it? Also when I am passing the list i don't need to worry about log of 0 is that going to through out program-breaking error or just warning because i am converting the output of my likelihood function to just append each time the logged site prob instead of accumulating and should i cast to np.array for making faster but i don't make any subsequent operations so not needed right because i am doing list comprehension for subtraction but is it more efficient another way? When I sort values using the `.sort_values` and I want to replace the indexing is there another command `drop_index` or something?

When doing enumerate function on a list does that return two values the i index and val? And i think to reduce number of calls instead of recalculating likelihood everytime can i just augment the functions such that it can accept both list and number (number because i am going to send just 1 sometimes as a parameter) and list because for each codon site there will be a different parameter value associated. Would that be efficient since then only 2 calls and i plan on just again np.array cast the list resulting from the likelihood call with the df[omega].values passed in minus the array with just 1 passed in as omega? Also i want to make a y=x line with the fitted values i have. Is it possible to do simply without correlation or i was thinking just have at one end the start so where the minimum point in the graph and i have points in a list so can use np.min i think and since I am plotting two points min of those i believe right and then the max at the top right does that work or do I have to do as a correlation. In other words i am thinking of doing like first np.min(df2["dLnL"].values) and np.min(our_dLNL)) so i get minimum and then take the minimum across both as the min point and so same for max. Then so from the plot documentation its says its two lists the one list is the x points and the other is the y points right so should it be just [min, min], [max, max] so that its connecting points (min,min) to (max,max) right? What is the difference between min and np.min, is np.min min over an array where as min is two scalars? Finally I used the axhline function from documentation to make a horizontal line at 0 but its solid - is that the correct scientific figure or should i choose different option for line?" The AI told me that there is an isinstance and I can compare it to the np.ndarray. The AI approved of my rudimentary y=x line but clarified that it should be [min, max] and [min, max], which another person then expanded on for the linear regression and then for per site plot with peaks labelled. The AI approved the vectorization I was doing for my likelihood function as well as gave me an explanation for the axhline function in python giving me color and line suggestions. The AI also confirmed to me that there is a reset_index and that I should use it and set drop to be True as well as gave me suggestions on the benefits of using the np.array cast to avoid the current list comprehension work I was doing. The implementation still is my own work.

Shubham Mohole. I used Gemini (December 13-20, 2025) with the following same prompt while conducting literature review: "Can you take this source information and turn into bibtex citation for me and also will you tell me if there is more well-cited literature in the area or is this the standard citation? [insert source]". The AI helped me for sources that did not have a downloadable bibtex citation to convert the source information I had gathered into a latex artifact that I could use for the bibtex citations. I used this in the bibliography and also the AI, since I was using Google Scholar with keywords, at times pointed me to better more cited literature on the terms I was searching for.

Shubham Mohole. I used Gemini (December 13, 2025) with the prompt: "I have tables from the reproduced results. Could you give me in latex template for a tables simple just a table that shows like the values I got below are close to ones that were reported in the paper so just like a side by side so like i want rows as models three of their models and then just do a comparison of our three so the columns should be like LL (our reproduction), LL (reported in paper), beta parameter (our reproduction), beta parameter (reported in paper), omega parameter etc. and have the metric as like a top category and then maybe a bar and then the reported vs reproduction. Just give me the table latex and I can fill in." The AI gave me a template for the tables to use but I opted to just use the version of the table found in the template latex file and I manually filled in the values from our modelcomparson.md files. The tables are my own work.

Shubham Mohole. I used Gemini (December 18, 2025) with the prompt: "I want to confirm my implementation/idea for an entropy vs delta log likelihood graph. So i have a list of the amino acids columns which is in the dms data so I am going to have that list and then i found the scipy entropy function. Now i believe all i have to do is call the entropy function but for the axis i think is should set to 1 so that its row by row and i should define the columns because dont want it to be caught by the the site index right? Also if I want the dots to have varying colors i tried with color but it does not work and documentation says color is color mapping so do I modify this parameter by the optimized omega I pass in c=? Also is the command for the bar as a legend for the coloring the colorbar? Also i see a lot of noise i believe in the site by dLnL plot i have so i am thinking of another figure which takes the average across 15 sites so i see where the spikes are really happening so i saw that there is a rolling function but when i used my array.rolling there was an error. Do pandas functions not work on numpy arrays or what?" The AI confirmed my implementation approach and explained the parameter c= I needed to fill in and differences between arguments of colors vs c. I had made a mistake in passing in just the df2["omega"] which is just a series instead of the values array. The AI also told me I would need to cast my np array into a pd.Series to then calling the rolling and then call mean function. The implementation is still my own work.

Evan Cui. I used Gemini (Dec 12, 2025) with the prompt. "What Python libraries can help me convert a phylogenetic tree with nodes having three children into a tree with at most two children per node?" The AI suggested the ETE (Environment for Tree Exploration) Toolkit's resolve_polytomies() function as a way to do this. The specific implementation of the function is still my own work, as I had to create the tree after getting the newick string input and write the output to a new file after using the toolkit's function.

Note:

- This is the title page (required). It does NOT count toward the minimum page requirement.
- In the main text, do NOT adjust margins, font size, character spacing, line spacing, or add blank lines to artificially increase the number of pages to meet the minimum page limit. Doing so will result in a significant point deduction.
- Please read the “Final Project Report Guide” on Canvas carefully.
- Each group member must submit the “Author Contribution” form individually. The template is available on Canvas. Please note that, while the final report is a group submission, the author contribution form is an individual submission. Failure to submit the author contribution form by the due date will result in a 1-point deduction from the final report grade.

1 Introduction

1.1 Biological Motivation

Viruses like HIV and Influenza, face a number of challenges to survive in human populations. One approach to ensure these viruses take to survive is changing their surface glycoproteins, HIV envelope (Env) and Influenza hemagglutinin (HA), thus avoiding immune surveillance [1]. These proteins are specifically targeted by neutralizing antibodies, which, as a result, impose a strong selective pressure for mutation. Nevertheless, HIV and Influenza are not just shelled by the antibodies; in addition, viral evolution is limited by the need to keep the structural integrity for host cell entry. Hence, changes that bring immune avoidance but hinder cellular entry are ultimately deleterious. These counteracting forces are the reason why viruses must walk along a thin evolutionary ridge between immune avoidance and function.

In this study, we focus on two well-characterized HIV Transmitted-Founder strains, BF520 and BG505. Transmitted-Founder viruses are the variants that can successfully infect new hosts. This makes them useful for understanding how viruses evolve long-term under immune pressures [2]. BF520 and BG505 have similar structures, but differ at over 100 amino acids (AA) [3], providing substantial genetic divergence for comparison. Moreover, the extensive study of both strains allows our observations to be interpreted with the context of existing literature.

Despite the similarity of the immune pressures HIV and Influenza fight with, they use different methods for evolution. Influenza has evolved mainly as a result of the process termed strain replacement, a process under which new strains take over older ones in the population [4]. While Influenza has a brief existence in its host, HIV can continue to exist in particular host cells for a long time [5]. For enduring the immune system, HIV deploys more ingenious techniques such as a configurational mask and an extensive glycosylation pattern that can cover itself from the antibodies that bind to its conserved areas [6]. The above reasons are why HIV is a particularly difficult model as the immune evasion may be a result of even trivial AA changes.

1.2 Modeling Motivation

A useful way to explore these selective pressures is using deep mutational scanning (DMS). DMS looks at the fitness impacts of nearly all possible changes across a protein's sequence [3] and has already been used to model the evolutionary path of H3N2 Influenza mutations [7]. DMS produces site-specific amino acid preference profiles, which demonstrate the limits imposed by structural and biochemical factors. The maps reveal the variability field at different locations: some sites undergo only a limited number of substitutions, while the others are more permissive. Hence, DMS becomes the tool for an examination of the viral fitness landscape.

Standard phylogenetic models such as Goldman M0 and M5 attempt to capture the selective pressures by averaging the rates of substitutions or by fitting them into statistical distributions [8]. Although this can bring out the overall trends, it does not deal with the local constraints. One limitation of these models is the inability to differentiate whether the conserved sites are essential for the function, like buried core residues or cysteines forming disulfide bridges. Therefore, conventional models can misjudge purifying selection as neutrality, failing to notice the multiple constraints imposed by the structure of the proteins.

This difference is substantial when studying viral glycoproteins which have both highly conserved and highly variable regions. Highly-conserved sites are often needed for functions such as protein stability or receptor binding, with mutations in these regions typically resulting in a loss of function. In contrast, variable sites are more amenable to a wider range of AA substitutions and are often targeted by mutations aimed at evading the immune system. For instance, the regions of high variability in the Env protein of the HIV virus help the virus to exit from antibodies while the underlying core remains intact [6].

The main challenge we are facing is that classical phylogenetic models do not take into account site-specific biological constraints and therefore, lack the ability to describe how viruses evade immune pressure and, at the same time, preserve their key functions. For this, we are employing an experimentally informed codon model (ExpCMs) which captures site-specific amino acid preferences resulting from DMS data by integrating them into the models. ExpCMs explicitly encode the structural and fundamental constraints of experimental fitness data left out by standard models by anchoring inference to experimental fitness landscapes. Additionally, the framework facilitates selection model differentiation through which we can determine the spots where the natural viral evolution diverges from lab-measured preferences for the linear paths thus, highlighting the positions with the additional pressures *in vivo* [3]. By employing these techniques we can achieve a biologically more realistic comparison between the natural evolution and the experimental expectations; in this way, we can get insight into how HIV Env manages to escape immunity with functionality at the same time across different genetic backgrounds.

Given the problem that standard phylogenetic substitution models ignore site-specific constraints at different regions of the protein encoded by the gene and the biological motivation outlined above in relation to HIV and Influenza, our contribution is to re-implement the ExpCM likelihood function to compare lab DMS data to selection process in nature for HIV and Influenza. Three sub-questions we aim to address in this report are:

SQ1: How do the differences between Influenza and HIV manifest through metrics such as GC content, sequence length, and mutational tolerance?

SQ2: How accurate was our re-implementation compared to the reproduced PHYDMS gold-standard implementation?

SQ3: For which sites or functional domains does utilizing the DMS data highlight external immune pressure?

2 Methods

2.1 Alignments, Deep Mutational Scanning Data, and Preprocessing

For the HIV analysis, we used the sequence alignment and DMS data found from the Haddox et al. [3] HIV paper. We adapted the analysis_notebook.ipynb found in the paper's accompanying GitHub [repository](#). The notebook utilizes the DMS_TOOLS2 and BIOPYTHON packages. In the repository, there were already the FASTA files for the Env coding sequences of the two homologous strains, BF520 and BG505. The following are the steps the notebook took and the descriptions in this paragraph are based on the comments made in that file by the original authors of the notebook. The notebook defined the codon sites for both strains that were mutated, which were 31-702 with respect to the HIV reference strain of HXB2 numbering. The notebook reads a manually edited alignment file containing the aligned sequences of the Env homologs and HXB2 generated by MAFFT. The manual adjustments were made in the variable loop sites. After reading the alignment, the notebook generates a dictionary which pairs numbering of the homologs to HXB2 numbering. Afterwards, using the plasmid DNA library, which are the circular pieces of DNA which are the results of different versions of the Env gene that were mutated, and the passaged virus library which store the results of having forced the plasmids in cells to then see if virus would be duplicated via those cells, with the DMS2_BATCH_PREFS function, the notebook calculates amino acid preferences at every site to generate the DMS data. For the alignment file, the notebook downloads data the HIV1_FLT_2016_env_DNA.fasta file from the Los Alamos HIV database, extracts the Clade A sequences, cleans them, and then adds the BG505 and BF520 lab sequences. Then, to define the mask, they noted sites like signal peptide and cytoplasmic tail sites because there is no DMS data, sites in variable loops that had too many insertions and deletions, and gap sites. Using this mask, the notebook then deletes the columns in the existing alignment file. This step provided the DMS data for both strains and the resulting masked alignment file. The modifications needed to the existing source notebook were the removal of the plotting cells due to package conflicts, the editing of the dmstools2 utility functions for renumbering sites due to package errors, and other function substitutions from installed libraries. For Influenza, the alignment and DMS files were obtained from the Lee et al. [7] Influenza paper [github repository](#). The H3N2 human Influenza hemagglutinin alignment and DMS data based on the A/Perth/16/2009 strain were already provided so no preprocessing was needed. To summarize, the sequence files we used for HIV and Influenza were: the masked_alignment.fasta outputted by the Haddox et al. [3] analysis notebook and the H3_human_alignment.fa found in the Lee et al. [7] github repository. The HIV DMS data used was the BF520.csv and the BG505.csv, which was a result of running the notebook. The Influenza DMS data used was the summary_avgprefs.csv found in the Influenza github repository.

2.2 Phylogenetic Tree with RAXML-NG

The Haddox et al. [3] and Lee et al. [7] papers used RAXML [9] to generate the newick tree associated with the sequence alignment. Given this project's goal of obtaining the most accurate, statistically likely tree topology based on the alignments, we chose RaxML-NG (Random Accelerated Maximum Likelihood Next Generation) [10]. Using our sequence alignments (masked_alignment.fasta and H3_human_alignment.fa), we used the RaxML-NG package with the model set to the General Time Reversible with distributed gamma rate heterogeneity model [11]. This model was chosen because it was suggested in the repository for the package as the best model to use for tree inference on DNA alignment as per the Kozlov et al. [10] [repository](#), and with the gamma distribution as implied by the ExpCM documentation, it accounts for different sites in DNA sequence evolving at different speeds.

2.3 Phydms Reproduction

Using the PHYDMS software package on the BIO-HPC cluster, the phydms_comprehensive function [12] was run, which fit the experimentally informed codon model as well as the baseline YNGKP_M0 model for both the HIV files, and then separately the same function was run on the Influenza data. The pipeline outputs the model parameters which has the optimized global parameters, global log likelihood, and an omegabysite.txt which contains the delta log likelihood (dLnL). This last omega-by-site file for each model contains the by-site likelihood value, but is computed as the difference between the local optimal omega fit parameter (a term that will be discussed in section 2.4) and the null model where omega is set to 1. The results in these files served as the gold standard for the re-implementation of the likelihood function. To ensure reproduction was accurate a comparison was made with the values documented by the papers in their respective repository and Figures results, as seen in Table 1 and Table 2.

2.4 Re-Implementation

The following descriptions of the re-implementation are sourced from the Hilton et al. [12] and its accompanying numerical implementation [website](#). In our model, we first implemented equations 1-3 of ExpCM, which make up the ExpCM substitution model. We chose to keep these three equations in linear space instead of log space, as they do not result in the underflow that log likelihoods would help avoid, but we began using log likelihoods instead in the Felsenstein's algorithm implementation. This substitution model is a mutation-selection model, which means it views changes in DNA sequences as being the product of the probability of mutation and fixation probability [13]. This corresponds to the first equation of the substitution model, which is defined as:

$$P_{r,xy} = \begin{cases} Q_{xy} \times F_{r,xy} & \text{if } x \neq y, \\ -\sum_{z \neq x} F_{r,xz} Q_{xz} & \text{if } x = y. \end{cases}$$

The equation calculates the rate of substitution from codon x to codon y at site r by multiplying the non-site-specific rate of mutation from x to y with the selection on this mutation specifically at site r if codon x and codon y are different. If codon x and codon y are the same (which corresponds to diagonal entries of the transition rate matrix P_r), the term equals to the negative sum of all other terms in the row. This is expected because the rows sum to zero in a transition rate matrix, as discussed in lecture 12.

The non-site-specific rates of mutation Q_{xy} are calculated in equation two, which takes as input the transition-transversion ratio κ and three nucleotide frequencies ϕ_A, ϕ_C, ϕ_G as input (since nucleotide frequencies must sum to one, we can deduce ϕ_T by subtracting the sum of the three other frequencies from one). Equation two is defined as

$$Q_{xy} = \begin{cases} \phi_w & \text{if } x \text{ is converted to } y \text{ by a single-nucleotide transversion to } w, \\ \kappa \phi_w & \text{if } x \text{ is converted to } y \text{ by a single-nucleotide transition to } w, \\ 0 & \text{if } x \text{ and } y \text{ differ by more than one nucleotide.} \end{cases}$$

In other words, if codon x can be converted to codon y by a single transversion of a nucleotide to w , Q_{xy} equals to ϕ_w . If codon x can be converted to codon y by a single transition of a nucleotide to w , Q_{xy} equals to $\kappa \phi_w$. Mutations with more than one nucleotide change have zero probability because the creators of ExpCM chose to use a HKY85 model to calculate Q_{xy} , and the HKY85 model calculates the rate of mutation in an infinitesimal time interval, meaning that the likelihood of multiple mutations occurring in that interval is negligible. Thus, $Q_{xy} = 0$ if codon x and codon y differ by more than one nucleotide.

The site-specific selection terms $F_{r,xy}$ are calculated in equation three, which takes as input the preference of site r for amino acid a $\pi_{r,a}$, stringency parameter β , and rate of non-synonymous to synonymous mutations ω . Define $\mathcal{A}(x)$ and $\mathcal{A}(y)$ to be the amino acid encoded by codon x and codon y , respectively. Equation three is defined as:

$$F_{r,xy} = \begin{cases} 1 & \text{if } \mathcal{A}(x) = \mathcal{A}(y) \\ \omega & \text{if } \mathcal{A}(x) \neq \mathcal{A}(y) \text{ and } \pi_{r,\mathcal{A}(x)} = \pi_{r,\mathcal{A}(y)} \\ \omega \cdot \frac{\ln((\pi_{r,\mathcal{A}(y)})^\beta / (\pi_{r,\mathcal{A}(x)})^\beta)}{1 - ((\pi_{r,\mathcal{A}(x)})^\beta / (\pi_{r,\mathcal{A}(y)})^\beta)} & \text{otherwise.} \end{cases}$$

In other words, if codon x and codon y encode the same amino acid, $F_{r,xy}$ equals to one. If codon x and codon y encode different amino acids and site r has the same preference for either amino acid, $F_{r,xy}$ equals to ω . Otherwise, $F_{r,xy}$ equals to $\omega \cdot \frac{\ln((\pi_{r,\mathcal{A}(y)})^\beta / (\pi_{r,\mathcal{A}(x)})^\beta)}{1 - ((\pi_{r,\mathcal{A}(x)})^\beta / (\pi_{r,\mathcal{A}(y)})^\beta)}$.

After implementing the ExpCM substitution model equations, we then implemented equations 11-13 of ExpCM, which calculate and normalize the site-specific stationary state of the substitution model defined by $P_{r,xy}$ from equation one. The stationary state is needed to compute the transition probability matrix and is used directly in Felsenstein's algorithm when computing the likelihood at the root node. The equation for the stationary state (of codon x at site r) is defined as:

$$p_{r,x} = \frac{q_x f_{r,x}}{\sum_z q_z f_{r,z}}.$$

where

$$f_{r,x} = (\pi_{r,\mathcal{A}(x)})^\beta$$

and

$$q_x = \phi_{x_0} \phi_{x_1} \phi_{x_2}$$

where x_0 , x_1 , and x_2 are the nucleotides at the first, second, and third positions of codon x . In other words, the numerator of the stationary state for codon x at site r is the preference for the amino acid encoded by codon x at site r to the power of the stringency parameter β , multiplied by the nucleotide frequencies for each nucleotide in codon x . The denominator normalizes the frequencies by summing this product across all possible codons, meaning that the stationary codon frequencies at each site sum to one.

After implementing the normalized stationary state equation for the substitution model, we construct the transition rate matrix P_r for each site using equation 1 from above. We re-scale the branch lengths so that they are in units of substitution per site. The scaling is because the results from optimizing with the transition rate matrices are not in units of substitution per site. The probability that site r undergoes a substitution in a unit of time is defined as:

$$-\mu \sum_x p_{r,x} P_{r,xx}.$$

μ is a substitution rate which we set to one as a simplifying assumption because our project is not doing any work with the absolute speed of mutation, $p_{r,x}$ is the stationary state of codon x at site r , and $-P_{r,xx}$ is the sum of the substitution rates from codon x to codon y at site r . This is because $-P_{r,xx} = \sum_{x \neq y} P_{r,xy}$ by the construction of the substitution rate matrix for codon x at site r . So, we rescale the branch length by dividing the branch length resulting from optimizing using the $P_{r,xy}$ substitution matrices by $\sum_x p_{r,x} P_{r,xx}$.

After computing the transition rate matrix for each site, we can compute the transition probability matrix for each branch by computing $\mathbf{M}_r = e^{t\mathbf{P}_r}$, where $\mathbf{P}_r$ is the site-specific transition rate matrix and t is the branch length (time) over which the transition occurs. This exponentiation is significant, because it allows us to compute transition probability as a function over elapsed time, rather than a static quantity. We used the matrix exponential function from the SciPy library to compute $\mathbf{M}_r$.

After implementing our custom ExpCM model, we used Felsenstein's pruning algorithm to calculate the likelihood of a tree. We used log-likelihoods to avoid underflow errors, especially as our input sequence data had hundreds of sites. Because the input phylogenetic Newick tree may not be a binary tree, we used the ETE toolkit to produce a new tree that fixed any nodes with more than two children by splitting it into a series of branches with at most two children. We used the Phylo package from the Biopython library to read the new tree and return a postorder traversal over the nodes, as we could leverage other functions inside the package to assist and speed up our analysis of the results. We also reused the function from the homework assignments to read fasta files and output the sequences and size from the file. We decided to treat a site as being a codon instead of a single nucleotide like in the homework assignment, because phydms uses codons when implementing its formula to calculate the likelihood of a tree. The phydms equation for calculating the partial conditional likelihood of the subtree rooted at n is given as:

$$L_{r,n}(x) = \begin{cases} \delta_{xS_{r,n}} & \text{if } n \text{ is a tip/leaf node with codon } S_{r,n} \text{ at site } r, \\ 1 & \text{if } n \text{ is a tip/leaf node with a gap at site } r, \\ \left[\sum_y M_{r,xy}(t_{d_1(n)}) L_{r,d_1(n)}(y) \right] \left[\sum_y M_{r,xy}(t_{d_2(n)}) L_{r,d_2(n)}(y) \right] & \text{otherwise.} \end{cases}$$

where $\delta_{xy} = \begin{cases} 0 & \text{if } x \neq y, \\ 1 & \text{if } x = y. \end{cases}$, $M_{r,xy}$ is the transition probability matrix for site r from codon x to y , t_n is the length of the branch leading to node n , and $d_1(n)$ is the left child and $d_2(n)$ is the right child of node n . So, we see that the recurrence is very similar to the recurrence we learned in lecture, except that instead of considering four nucleotides in a site, we now must consider 61 possible codons for a site, i.e. at the leaf nodes, one out of 61 terms in the partial likelihood vector will be one and all the others will be zero. At the root node, we take the dot product of the site-specific stationary state with the partial conditional likelihood of the tree rooted at the root node to get the total likelihood at the site, and sum up the site-wise log-likelihoods to get the overall log-likelihood.

2.5 Downstream Analysis

Our first step was to reproduce locally the results using the gold-standard PHDYS package. Our motivation behind this and not just using the results reported in the HIV and Influenza papers is twofold. First, the Influenza repository does not, to our knowledge, contain a copy of their phdys results. Second, the results reported in the papers because they are from 2018 may use, now considered, outdated versions of key optimization packages for the maximum likelihood optimization that PHDYS relies on. For instance, SCIPY, in our reproduction run, as per the ENVIRONMENT.YML file in our repository, is set to

SCIPY=1.10.1, a version that was released in 2023 [14]. For these two reasons, we decided to reproduce the gold-standard. As expected our reproduction outperformed the reported values as seen in Table 1 and Table 2 for both HIV and Influenza for both the ExpCM models and even the baseline YNGKP_M0. We decided not to compare against the M5 baseline because, in Haddox et al. [3] results, the only model that performed better than the M5 baseline was a omega from gamma distribution variant of the standard ExpCM model, which will be touched on in the Discussion section. Thus, for the purposes of analyzing our re-implementation we compare the reproduction values.

Next, it is important to justify using both HIV and Influenza by showing their differences as per **SQ1**. We conducted a comparative summary statistic analysis across commonly used metrics for genes. We used the BIO.SEQIO package to parse the FASTA files. This allowed us to get total number of sequences in the file, alignment length, percentage of Guanine and Cytosine bases, and gap percentage. We also counted how many sites were in the DMS data for both HIV and Influenza. This resulted in our Figure 1.

To address **SQ2** and to create the rest of the plots in the Figures section, we called our re-implemented likelihood function. For Figures 2-4 our goal was to verify that our re-implementation was successful against our reproduced results to answer **SQ2**. For Figure 2 we wanted to make sure our computed global likelihood was similar to the reproduced global likelihood when running the PHDYMS package. For Figure 3, we wanted to ensure that our maximum likelihood occurs at the expected beta parameter value when we hold all parameters constant and change the beta parameter with steps of 0.2 from 1.0 to 5.0. For Figure 4, we decided to create our own omegabysite file similar to that produced by the phdymms reproduction. We calculated the log likelihood difference between our re-implemented likelihood function with the optimized parameters and our re-implemented likelihood function with the null model parameter which we define as passing in the omega parameter as 1 (dLnL). The dLnL information was gathered based on the comments left in the omegabysite text files after the PHDYMS outputted its results but it is a simpler reduction of the null model that PHDYMS implements which we will discuss as part of our limitations in the Discussion section. To validate Figure 4, we decided to do a linear regression using the SCIPY.STATS.LINREGRESS to see if our points fit a $y = x$ line and calculate the R^2 .

Moving toward more biologically-informed Figures after our re-implementation was validated and to answer **SQ3**, we first plotted our dLnL against the sites and cross-referenced with sites reported in the respective papers. We wanted to see if we had similar peaks in likelihood at those specific regions. Instead of manually labeling outlier points, we used the SCIPY.SIGNAL.FIND_PEAKEs function with threshold of prominence at 2.0 to find the top 10 local maxima. For our Figure 6, we utilized the python package scipy again using its ENTROPY function, which calculates the Shannon Entropy across the 20 amino acids in the DMS data. Specifically, for site i , the Shannon entropy value according H_i to Lee et al. [7] is calculated as $H_i = -\sum p_{i,j} \log_2(p_{i,j})$. H_i can range from 0, where only one amino acid is tolerated at that site, to $\log_2(20) = 4.3$, where each of the 20 amino acids are equally tolerated. A point high in dLnL (vertical) indicates the model selecting strong functional signals, and if those same points are yellow, then this indicates agreement with the DMS data about intense selection pressure. The most biologically important sites on each gene are thus in the top right with high entropy, which in the context of viruses often correspond to surface proteins that rapidly evolve against immune response, and the top left with low entropy, which correspond to proteins that lose function with any mutations. For Figure 7, we aimed to visualize the difference between protein domains for Influenza. Using Wilson et al. [15] we partitioned the sequence between the HA1/Head, sites 1–328, and HA2/Stalk, sites 329+. Using the dLnL, we plotted a violin plot to measure the probability density as well as to identify the outlier trends between the head and stalk.

3 Results

Figure 1: Figure 1 highlights the computed summary statistics on the dataset, which includes both strains of HIV and Influenza. For the number of sequences in FASTA files, the Influenza data has 100 sequences, which is the most. The two HIV strains have equal sequences, alignment length, guanine/cytosine content in bases, gap percentage, and number of sites in respective DMS data. The Influenza data has a slightly decreased alignment length compared to HIV, a higher percentage of bases that are guanine or cytosine, and a slightly decreased number of sites in its DMS data. For the gap percentage, it is nearly zero while the HIV data have a gap percentage of approximately 2.9.

The gap percentage for Influenza being close to 0 compared to the 2.9 percentage for HIV showcases how the HIV evolves more rapidly. It is more flexible than the Influenza virus, as gaps allow for insertions and deletions. This observation ties back to the initial motivation behind studying both HIV and Influenza. Another result from the Figure is that HIV has lower GC percentage than Influenza.

Figure 2: Moving on from the summary statistics, Figure 2 aims to evaluate whether the likelihood values generated by the re-implementation were similar with the log-likelihood generated by phdymms, given the optimized parameters reported by phdymms.

The re-implementation had success in mimicking the observation of ExpCM performing better than the YNGKP baseline. As reported in Figure 1, our likelihood of -36,734.3 (for BF520) and -37,345.2 (for BG505) is significantly better than the Goldman Yang M0 model at -39,909.3 based on the reproduction using the phydms package as per Table 1. Haddox et al. [3] employed a Likelihood-Ratio Test for statistical significance. The equation is as follows: $LRT = 2 \times (\ln LL_{ours} - \ln LL_{baseline})$ [16]. Substituting our values we get $LRT = 6350.1$. To compare this to chi-square distribution, we find the difference in the number of free parameters. The Haddox et al. [3] paper mentions that the M0 variant has 11 parameters and ExpCM has 6 parameters itself. Thus the degree of freedom is 5 and at 0.5% statistical significance the critical value is 16.750 so the improvement over the baselines is significant [17]. Similarly, for Influenza, the re-implementation result of -8,470.6 was better than the M0 at -9702.3 as per Table 2 reproduction column. In regard to accuracy compared to the phydms package run for the same model, for Influenza the re-implementation (-8,470.6) was slightly worse than the (-8,437.9). The HIV re-implementation was also slightly worse: for BF520, the re-implementation resulted in -36,734.3, whereas the phydms run outputted -36,136.6. For BF505, it was -37,345.2 compared to the phydms reproduction at -36,850.9. The percentage difference is calculated as less than 1.65 percent considering both HIV and Influenza. In log scale 1.65% is a substantial difference demonstrating a limitation of our re-implementation, however, as discussed in Figure 3, the maximum likelihood varying the stringency parameter aligns with the gold standard phydms implementation suggesting the landscape is comparable. Also, our re-implementation performs better than the values listed in the Reported column in Table 1 and Table 2 for the ExpCM model which suggest that our key algorithm implementation is correct.

Figure 3: Figure 3 tested the log-likelihood in terms of parameter values for HIV BF520, HIV BG505, and Influenza models. As expected, the reimplemented likelihood function reaches its maximum at the optimal beta parameter values, which is ~ 3.2 for HIV BF520, ~ 2.3 for HIV BG505, and ~ 2.4 for Influenza. The likelihood for all models follows a downward parabola that peaks at the optimal parameter. Comparing the peaks to the stringency parameters listed in the reproduction, it achieved similar peaks to what is reported as the optimal parameters for both HIV and Influenza. We also notice at both extremes of not following DMS results and of over-weighting the DMS data the likelihood suffers.

Figure 4: Figure 4 examined at the site-wise likelihood distribution, comparing the phydms dLnL to our dLnL. Each point is a codon site in the alignment. There is a clustering of points on positive side near 0 and this is expected as again the dLnL is a comparison of the ExpCM LL values minus the null model of $\omega = 1$. The ExpCM model performs better than the null model. For HIV and Influenza a $y=x$ line was fit on the graph and to see the line of best fit have slopes between around 0.75 to 0.9, suggesting a linear relationship as expected. Since the slope is slightly under 1, this indicates that the reimplemented function is underestimating the dLnL improvement for some sites. There are a few outlier points which we believe may be due to gap issues. According to the R^2 value, approximately 20-30% of the variance in our models is explained by the phydms models. The Influenza model has the highest R^2 value, at 0.3651. Again similar to the Figure 2 analysis such correlation does show a limitation but through this site-wise failure analysis we believe we may have low R^2 value due to numerical stability issues which was noted in the original documentation [12].

Figure 5: Figure 5 plotted the delta log-likelihood of each site. Some sites were local maximas in delta log-likelihood, and these peaks are labeled on the graph. There were 1-2 sites per model whose delta log-likelihood was drastically less. In regards to the peaks, first with the Influenza plot, there is a cluster of peaks ranging around 140 to 190. In fact, at site 144 (labeled as 143 due to indexing with 0-index array), as noted in Wiley et al. [18] in Table 1, residue 144 is the antigenic site A and is the place where antibody binding occurs and is a site that undergoes mutations to escape immunity pressures, which the null model fails to capture. ExpCM captures this resulting in the delta likelihood peak. In regards to HIV, the Haddox et al. [3] paper notes there are significant shifts at sites 505, 512, 516 due to changes in protein stability. This is reflected in the multiple peaks in the HIV 520 graph near the 500 range up to 516. The paper makes a comparison that BG505 is more stable than BF520 [3] and that is also reflected as in those same regions, the peaks have lower magnitude.

Figure 6 Similar to Lee et al. [7], Figure 6 plots mutational tolerance against the model improvement. By calculating the Shannon entropy/relative entropy and plotting that against the dLnL it is clear for all cases (the two strains of HIV) and Influenza there is a slight upward positive correlation. This reinforces the hypothesis that for low entropy sites where there is less flexibility leveraging the DMS data does not provide significantly more predictive power whereas for high entropy sites the null model suffers since it assumes homogeneity. Moving on, there is a difference between Influenza and HIV based on the scale of the dLnL scale of the scatterplots since HIV ranges near 20 whereas Influenza is less than 10 even at the max outliers. This reinforces the hypothesis that there are more flexible variable loops in HIV rather than Influenza for which the ExpCM model can perform better on using the amino acid preferences at those highly variable sites. The baseline model assumes that at these sites, since they are accepting many mutations they are neutral, setting ω to around 1, but it still can be functionally constrained [3]. The ExpCM model separately considers the amino acid preferences (DMS) and then the selection rate [3]. Finally looking at the outliers we see they are yellow which, corresponding to our color bar, are high ω values and ω ga

is the nonsynonymous over synonymous substitutions rate after considering amino acid preferences and so we see that our ExpCM model, correctly, is able to model these sites that are undergoing immune pressure.

Figure 7: Since Influenza HA contains two core regions, head (HA1) and stalk (HA2) [15], we wanted a region-level view of selection. To do this, we summarized the sitewise delta log-likelihoods from each ExpCM fits by region and visualized their distributions with violins. HA1 has a much broader spread with heavier tails, indicating stronger positive selection. Based on Lee et al. [7], our figure and observations are consistent with periodic, heterogeneous selection across different head sites and are likely a result of immune pressures. In contrast, HA2 is more tightly centered around zero, indicating more conserved evolution with fewer sites showing a strong divergence from a neutral model. This asymmetry between regions supports our hypothesis that adaptive changes cluster in the head, while the stalk remains more comparatively conserved.

4 Discussion

To address **SQ2**, our re-implementation of ExpCMs has been validated by successfully reproducing the log-likelihood peaks and optimal β parameters reported by Haddox et al. [3] and Lee et al. [7]. Indeed, these beta parameters offer an interesting insight into the stringency between lab-predicted and natural selection rates. Our β parameters are significantly greater than 1 in both reproduced papers (Table 1 and 2), suggesting that natural selection is more "demanding" than lab predictions. In other words, the high β suggests that a mutation with mild selective effects *in vitro* can be catastrophic or extremely beneficial *in vivo* due to harsher selection variables. Our results are consistent with gold standard results from Haddox et al. [3] and Lee et al. [7], indicating that our re-implementation successfully gauges the degree of the DMS data's signal in natural mutation predictions.

To address **SQ3**, our results also reflect the ExpCM's core benefit: that they can reflect site-specific biological variations from log likelihood contributions. ExpCMs, unlike conventional codon models, use the DMS data to distinguish between variable loops that are under intense pressure to mutate and structural cores that are under pressure to remain intact. This is well demonstrated by a spike in the site log likelihood for the Influenza virus at site 144 (Figure 5). This site is antigenic, i.e., where antibodies bind, so by spiking the log likelihood at that site, ExpCM can capture the natural variability present, unlike the null model. Similarly, there are multiple clusters in the HIV plot from sites 500-600, reflecting a structural core for the essential gp41 subunit [3]. This ability to predict biology through PhyDMS allows it to be transformational in viral medicine, allowing scientists to better predict which sites are more likely to mutate, thus showing which viral strains are more likely to persist and propagate, for vaccine development.

Indeed, not only are ExpCMs able to identify variable cores under high mutation pressure, but they offer increased predictive capabilities under high entropy compared to null models (Figure 6). Traditional models, like M0, which assume gene-wide homogeneity, consider high entropy sites as random, but ExpCMs can utilize DMS data to make more accurate predictions than M0. With this insight, ExpCMs understand that the high entropy sites are flexible in preference, but not wholly random. The dLnL is consequently even larger in high entropy sites than in low entropy sites, making ExpCMs a good fit for both.

Despite the broad use capability of ExpCMs evidenced by our re-implementation, there are multiple limitations that restrict its use case. Many of these limitations come from the nature of DMS data. Firstly, they are static, reflecting genetic preferences at a moment instead of historical evolutionary trends. They are also difficult to procure and assume independence between sites, which is not always true in practice. ExpCMs compute a 61x61 codon matrix per site, so they are not well-suited for genome-wide research, as they would be a very inefficient use of compute time and memory.

There are also re-implementation-specific issues, from not using the standard ExpCM without some of the additional add-ons. For example, our reproduction could have been improved by sampling the ω parameter from bins of a γ distribution. In addition, despite our work with log likelihoods to reduce underflow, we still observe a 2% difference in results and a low R^2 , because we did not follow PhyDMS's numerical stability work. Specifically, every few nodes during the tree traversal for every site, the algorithm prevents underflow by inflating all values in the current set in proportion to the local maximum. This would have added a layer of additional complexity, requiring correction of the data by summing the logarithms of all the scaling factors. Another limitation source specifically with the dLnL calculation is that the PHYDMS package performs a per site synonymous rate assignment for both null and the optimal models to distinguish then the mutation rate and selection pressure, so there would be a scalar multiplying for each site r in Equation 3 [12]. Given we did not implement the optimization component, consequently our null model is setting the $\omega = 1$ holding branch lengths constant which would imply that our synonymous rate for each site is fixed. This distinction we believe explains why our slope was slightly lower than 1 because of our simpler null model in calculating our dLnL per site, yet still even with this we were able to establish the intended linear relationship between our dLnL and the gold-standard dLnL.

Our results show several of these bottlenecks for ExpCMs. For example, BF520 and BG505 are both HIV-1 Env proteins, but

have vastly different β values of 3.2 vs 2.3. This indicates that genetic background or epistasis can alter a site's fitness, implying that rapidly-changing strains would need ExpCM results on an equally rapid basis, although they lack the ability to use prior probabilities for enhanced predictive power. Secondly, in Figure 4, our R^2 values ranged from 0.2 to 0.36, indicating that our re-implementation's ability to predict fitness differs widely from the original PhyDMS due to a high noise-sensitivity. Small issues with gaps or sequence alignments, especially in our re-implementation, can cascade into larger likelihood differences. Third, in Figure 6, as noted in the Results section there is a cluster of dLnL points near zero. This highlights that ExpCMs are not better than the null model at low-entropy sites. Finally, in Figure 3, there is a sharp U-curve for the β parameter, indicating that slight β differences can result in large log likelihood changes. However, we assume the same β across the protein, even though this may not be the case (with differing selection rates in different sites), which may over-smooth biological landscapes.

Given these specific advantages and disadvantages of the ExpCMs in our re-implementation, we conclude that they are best suited for rapidly-evolving viruses like Sars-CoV-2 with small sequence alignments available. Because ExpCMs perform well in high entropy settings and with high stringencies, they are well-suited for viruses that change quickly and are otherwise hard to predict with codon models with a numerical but not biological basis. In addition, for new, emerging strains of viruses with few sequences collected, DMS results fed into ExpCMs can create far more reliable results than traditional codon models like M5 that require large sequence alignments for statistical significance.

The next steps with our re-implementation to improve ExpCMs are to fix gap handling, epistasis, and optimization issues. As we noted in the Figure 4 analysis, our site-wise likelihood discrepancies are primarily caused by alignment gaps in the HIV-Env. Adding gaps in some way to our transition matrix or filtering our sites with over 20% gaps may resolve this issue. Secondly, we assume protein-wide uniform stringency. Allowing the stringency parameter to vary across different domains of the protein may help capture differences between, for example, the gp41 structural core and the gp120 variable loop in the HIV-env protein, and improve our R^2 value. Thirdly, we manually iterate over a range around the papers' optimal β parameter found by PhyDMS. As an extension we should include ω , κ , and ϕ , through a quasi-Newton method like L-BFGS-B to iterate over all variables simultaneously to ensure we are finding a true MLE (which is similar to what the original PhyDMS ExpCMs do). Finally, to solve the issue of epistasis, where sites are interdependent unlike what ExpCM assumes, we can use pairwise or contact preferences from co-evolutionary signals in our sequence alignment. That is, if two sites are adjacent or form connections, like disulfide bridges, we can consider the fitness of a mutation on one site to depend on the other.

References

1. Wyatt R, Sodroski J, 1998. The hiv-1 envelope glycoproteins: Fusogens, antigens, and immunogens. *Science*, 280(5371):1884–1888.
2. Goo L, Chohan V, Nduati R, Overbaugh J, 2014. Early development of broadly neutralizing antibodies in HIV-1-infected infants. *Nature Medicine*, 20(6):655–658.
3. Haddox HK, Dingens AS, Hilton SK, Overbaugh J, Bloom JD, 2018. Mapping mutational effects along the evolutionary landscape of hiv envelope. *eLife*, 7:e34420.
4. Smith DJ, Lapedes AS, de Jong JC, Bestebroer TM, Rimmelzwaan GF, Osterhaus ADME, Fouchier RAM, 2004. Mapping the antigenic and genetic evolution of influenza virus. *Science*, 305(5682):371–376.
5. Wei X, Decker JM, Wang S, Hui H, Kappes JC, Wu X, Salazar-Gonzalez JF, Salazar MG, Kilby JM, Saag MS, *et al.*, 2003. Antibody neutralization and escape by HIV-1. *Nature*, 422(6929):307–312.
6. Kwong PD, Doyle ML, Casper DJ, Cicala C, Leavitt SA, Majeed S, Steenbeke TD, Venturi M, Chaiken I, Fung M, *et al.*, 2002. HIV-1 evades antibody-mediated neutralization through conformational masking of receptor-binding sites. *Nature*, 420(6916):678–682.
7. Lee JM, Huddleston J, Doud MB, Hooper KA, Wu NC, Bedford T, Bloom JD, 2018. Deep mutational scanning of hemagglutinin helps predict evolutionary fates of human h3n2 influenza variants. *Proceedings of the National Academy of Sciences*, 115(35):E8276–E8285.
8. Yang Z, Nielsen R, Goldman N, Pedersen AMK, 2000. Codon-substitution models for heterogeneous selection pressure at amino acid sites. *Genetics*, 155(1):431–449.
9. Stamatakis A, 2014. RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, 30(9):1312–1313.
10. Kozlov AM, Darriba D, Flouri T, Morel B, Stamatakis A, 2019. Raxml-ng: a fast, scalable and user-friendly tool for maximum likelihood phylogenetic inference. *Bioinformatics*, 35(21):4453–4455.
11. Lavrov D, 2025. Computer Lab 4: Likelihood Analysis with RAxML-NG. https://isu-molphyl.github.io/EEOB563/computer_labs/lab4/raxml-ng.html. EEOB 563: Molecular Phylogenetics.
12. Hilton SK, Doud MB, Bloom JD, 2017. phydms: software for phylogenetic analyses informed by deep mutational scanning. *PeerJ*, 5:e3657.
13. Teufel AI, Ritchie AM, Wilke CO, Liberles DA, 2018. Using the mutation-selection framework to characterize selection on protein sequences. *Genes*, 9(8).
14. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, Burovski E, Peterson P, Weckesser W, Bright J, *et al.*, 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.
15. Wilson IA, Skehel JJ, Wiley DC, 1981. Structure of the haemagglutinin membrane glycoprotein of influenza virus at 3 Å resolution. *Nature*, 289(5796):366–373.
16. Evomics, 2025. Likelihood ratio test. <https://evomics.org/resources/likelihood-ratio-test/>. Accessed: 2025-12-17.
17. Watkins JC, 2025. Chi-square distribution table.
18. Wiley DC, Wilson IA, Skehel JJ, 1981. Structural identification of the antibody-binding sites of Hong Kong influenza haemagglutinin and their involvement in antigenic variation. *Nature*, 289(5796):373–378.

Figures

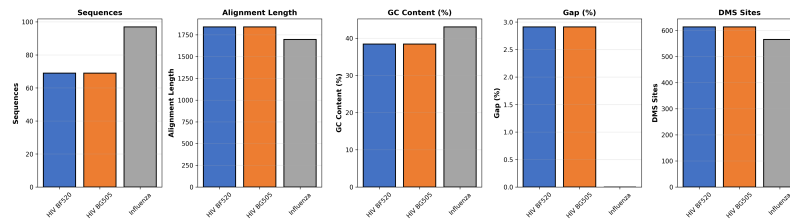


Figure 1. Summary Statistics between HIV strains and Influenza. Figure 1 shows statistics of the datasets from influenza and the HIV BF520 and BG505 strains which were used in the paper. The x-axis is the virus and the y-axis, in order from left to right bar graphs, is the number of sequences, the alignment length, GC (guanine/cytosine) in the form of percent, gap percentage and number of DMS sites using the viruses' sequence FASTA files and DMS data. Notably, the HIV and influenza datasets have important differences, as the presence of gap sites in HIV suggests it is more flexible or evolves faster than influenza, because gaps allow for insertions and deletions.

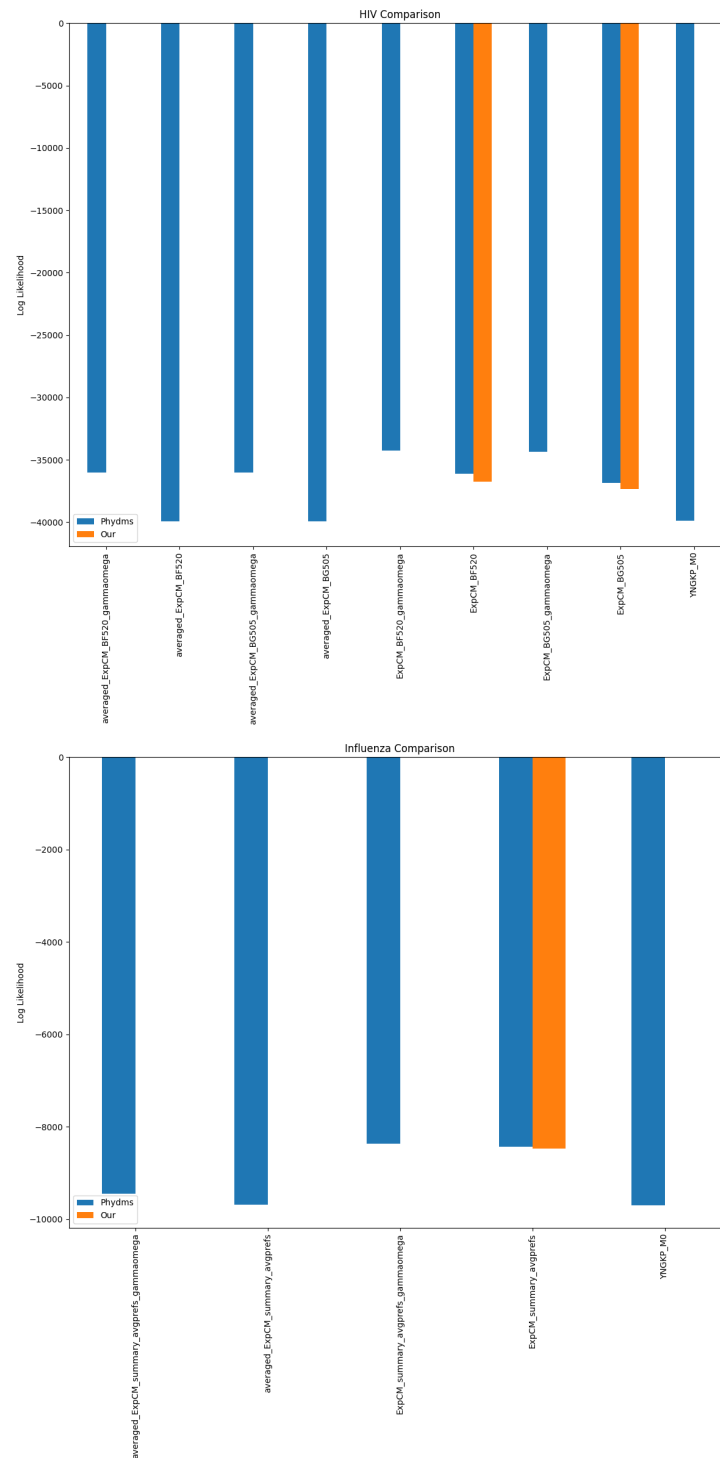


Figure 2. Model Comparison for HIV and Influenza. Figure 2 compares the log-likelihoods (the y-axis) between the phydms package with the log-likelihood from our implementation of phydms (the x-axis), given the optimized parameters generated from phydms. Our implementation performed 1.65 percent worse than the phydms package for both the HIV and influenza datasets. While this is a substantial difference in log scale, our implementation still has a significant improvement compared to models like YNGKP_M0 which are not site-specific.

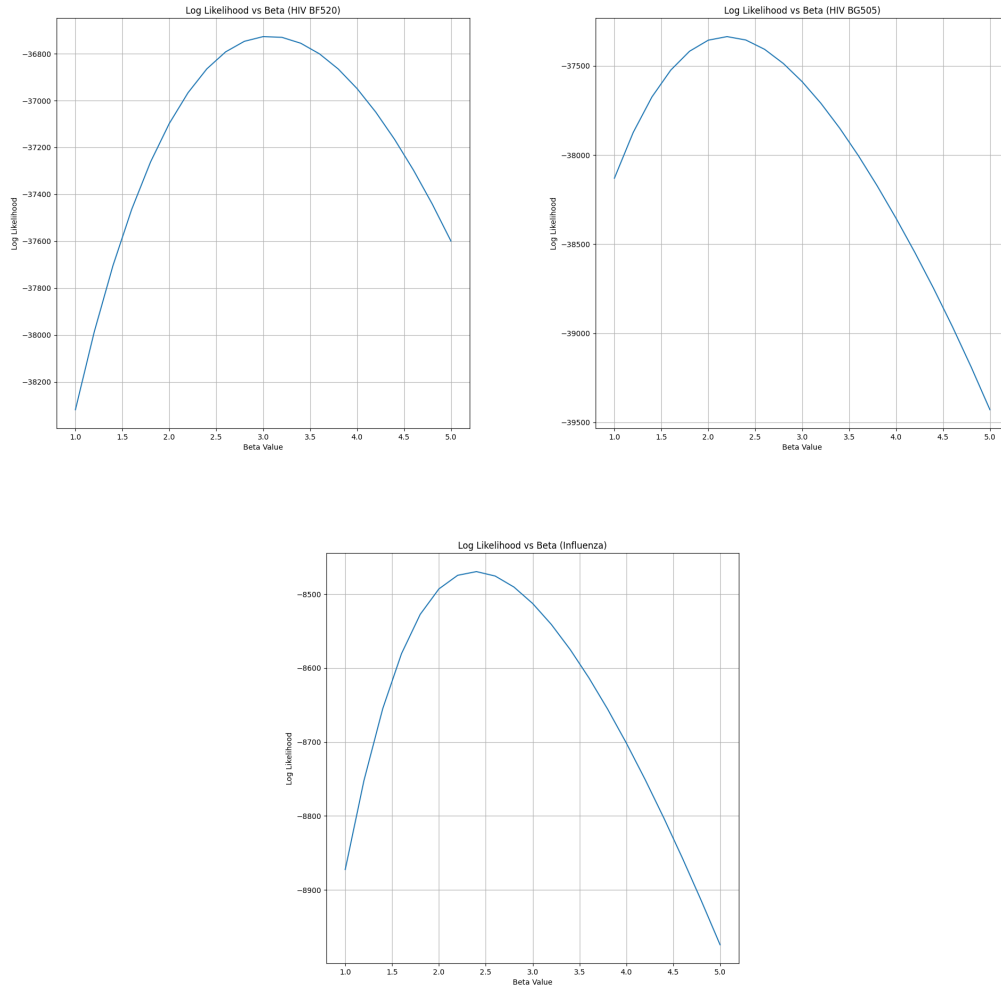


Figure 3. Likelihood Comparison Varying Beta for HIV and Influenza. Figure 3 compares the log-likelihood for different values of the stringency parameter β for HIV and influenza datasets, and plots the log likelihood on the y-axis and plots the value for the stringency parameter β on the x-axis. The figure confirms that the likelihood is maximized at the optimal value reported by phydm, which is ~ 3.2 for HIV BF520, ~ 2.3 for HIV BG505, and ~ 2.4 for influenza. The plots also show that overweighing or not considering the DMS data causes the likelihood to decrease, highlighting the usefulness of site-specific models.

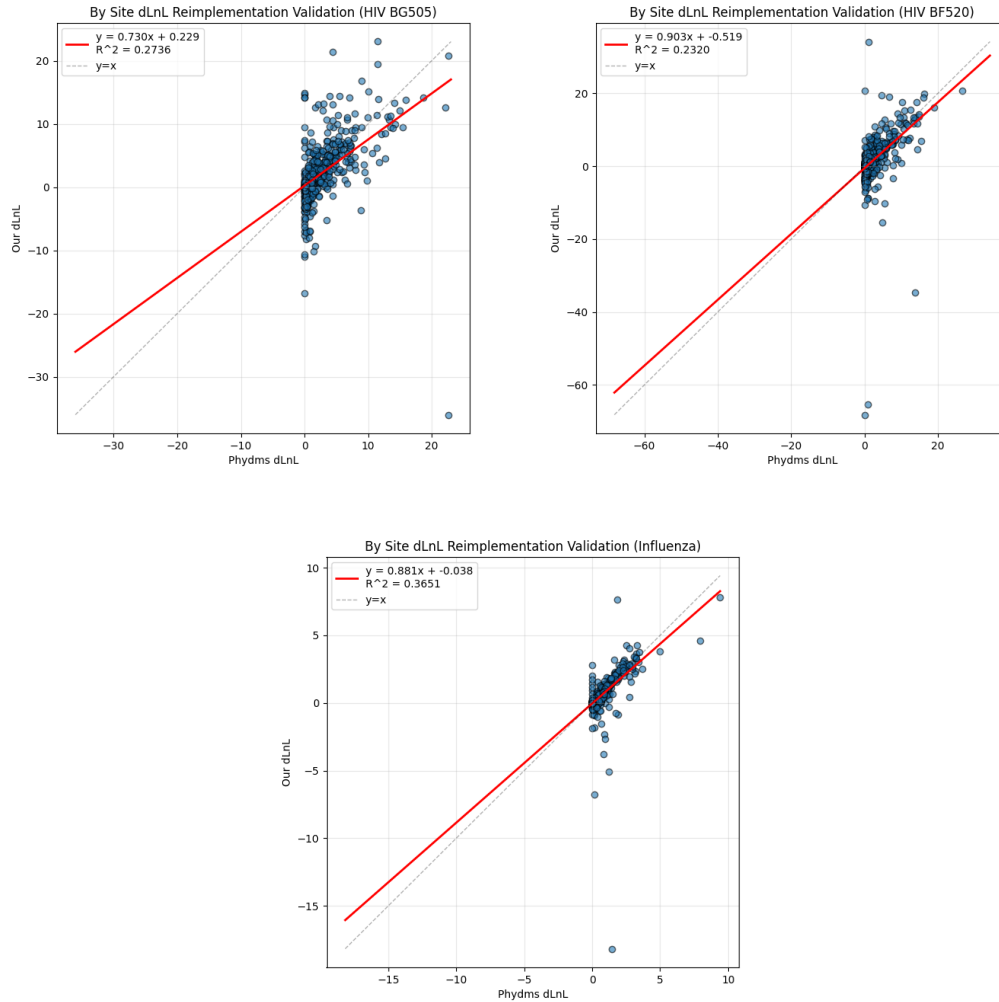


Figure 4. By Site dLnL Re-implementation Validation for HIV and Influenza. Figure 4 shows the distribution of the delta likelihoods across all of the sites for each of the datasets. The y-axis is our delta log likelihood which is calculated by subtracting the result of our re-implemented log likelihood function with the optimal parameters and the best fit omega by the re-implemented log likelihood function with optimal parameters but omega set to 1. The x-axis is the phdym's delta log likelihood calculated by subtracting the log likelihood that is derive from the optimal parameters and the best fit omega from their null model which substitutes omega for 1 as well as fits an optimal synonymous rate scalar value. Each point representing a site in the alignment is that site's (phdym's dLnL, our dLnL). The Figure plots the $y=x$ line in a dotted gray as an example of what the ideal result should be indicating perfect match between the phdym's site-wise dLnL and the re implementation dLnL. The red line showcases the line of best fit based on the actual data points. The main takeaway from this figure is that we achieved an underestimating linear relationship with said line of best fit for HIV and Influenza accounting for approximately 20-30% of the variance.

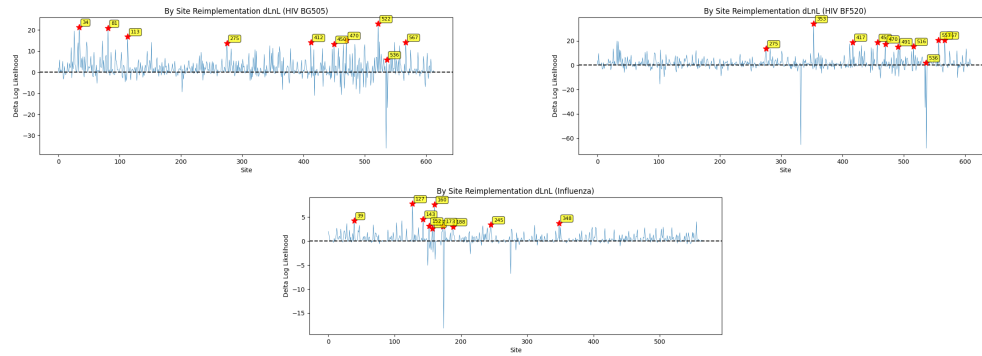


Figure 5. dLnL Site Likelihood Plot. Figure 5 shows the delta log-likelihood (dLnL) of each site by our re-implementation, which is calculated by subtracting the result of our re-implemented log likelihood function with the optimal parameters and the best fit omega by the re-implemented log likelihood function with optimal parameters but omega set to 1, for each of the datasets. The x-axis is the sites that are 0-indexed and the y-axis is the dLnL. The symbols on the top of some peaks are the local maximas in dLnL determined by SCIPY. Different regions, such as variable loops vs. conserved sites, have different log-likelihoods. In regards to the peaks, first with the Influenza plot, we see a cluster of peaks ranging around 140 to 190. In fact, residue 144 is the antigenic site A and is the place where antibody binding will happen and mutate which the null model fails to capture. In regards to both HIV strains, there are significant shifts at sites 505, 512, 516 due to changes in protein stability. This is reflected in the multiple peaks in the HIV 520 graph near the 500 range up to 516.

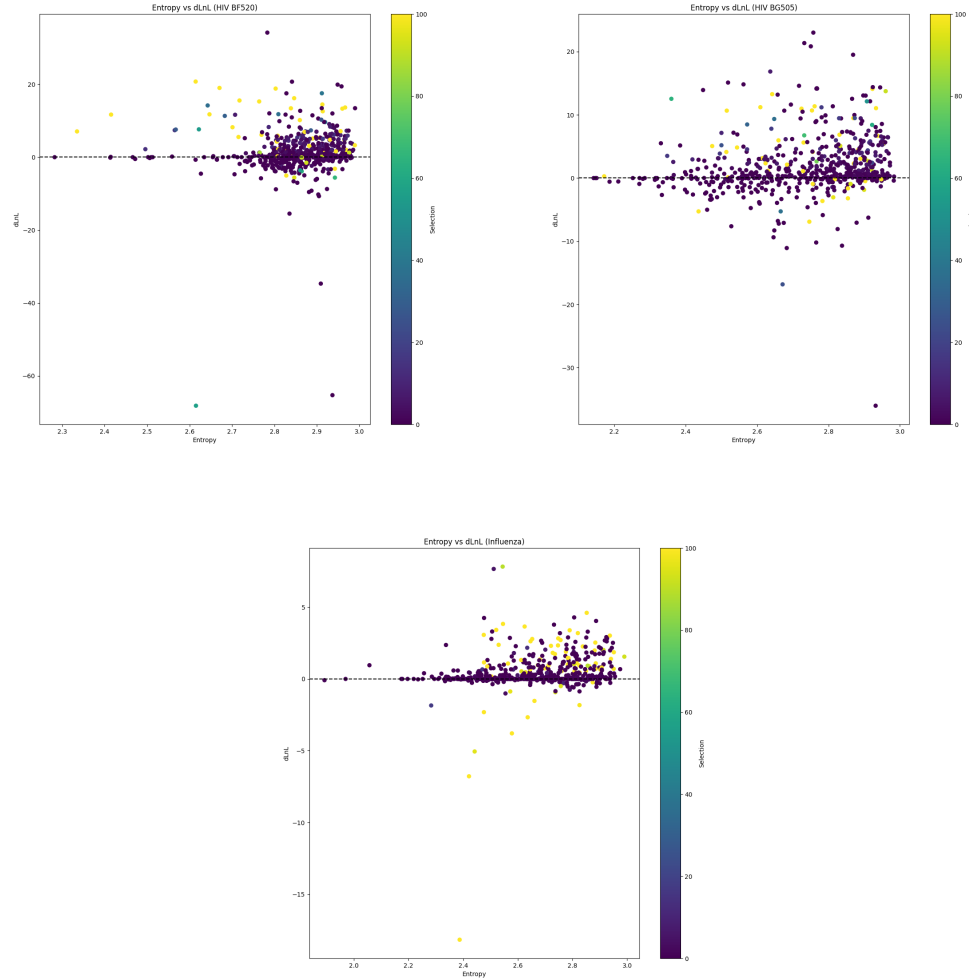


Figure 6. Entropy vs dLnL for HIV and Influenza The scatterplot compares the mutational tolerance (x-axis) to the model improvement represented by the delta log likelihood (y-axis), which is calculated by subtracting the result of our re-implemented log likelihood function with the optimal parameters and the best fit omega by the re-implemented log likelihood function with optimal parameters but omega set to 1. Sites are colored by their rate of nonsynonymous versus synonymous substitutions after considering site-specific preferences, (ω) parameter. By calculating the Shannon entropy/relative entropy (the x-axis) and plotting that against the dLnL we can see for all cases (the two strains of HIV) and Influenza there is a slight upward positive correlation. This reinforces our hypothesis that for low entropy sites where there is less flexibility leveraging the DMS data does not provide significantly more predictive power whereas for high entropy sites the null model suffers since it assumes homogeneity. Also, looking at the points that are above the 0 threshold we see they are yellow which, corresponding to our color bar, are high omega values so we see that our ExpCM model, correctly, is able to model these sites that are undergoing immune pressure.

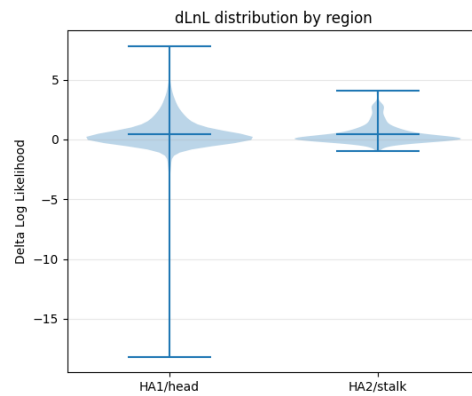


Figure 7. Influenza HA region dLnL Distribution. The above violin plot compares two different domains in the Influenza Hemagglutinin protein by plotting the delta log likelihood, which is calculated by subtracting the result of our re-implemented log likelihood function with the optimal parameters and the best fit omega by the re-implemented log likelihood function with optimal parameters but omega set to 1, for the head (HA1) and stalk (HA2). HA1 has wider spread indicating stronger positive selection whereas HA2 is centered around zero, indicating more conserved evolution with fewer sites showing a strong divergence from a neutral model. This asymmetry between regions supports our hypothesis that adaptive changes cluster in the head, while the stalk remains more comparatively conserved.

Tables

Table 1. Reproduced vs. Reported Model results for HIV The reproduced results were run with the Phydms package and values that are listed as reported are found in the tables of the Haddox et al. [3] HIV paper and their accompanying Github repository under the Figures folder and in the modelcomparison.md file.

Model	Log Likelihood		ω		β	
	Reproduction	Reported	Reproduction	Reported	Reproduction	Reported
ExpCM BF520	-36,136.6	-37,130.7	1.58	1.59	3.25	3.27
ExpCM BG505	-36,850.9	-37,934.9	1.32	1.32	2.34	2.33
YNGKP_M0	-39,909.3	-41,075.6	0.51	0.52	xx	xx

Table 2. Reproduced vs. Reported Model results for Influenza The reproduced results were obtained with the Phydms package and values that are listed as reported are found in the tables of the Influenza Lee et al. [7] paper.

Model	Log Likelihood		ω		β	
	Reproduction	Reported	Reproduction	Reported	Reproduction	Reported
ExpCM	-8,437.9	-8,441	0.91	0.91	2.47	2.47
YNGKP_M0	-9,702.3	-9,704	0.30	0.31	xx	xx