

ViralGeneClock

Tool for Estimating the Relative Mutation Rate of Different Genes across Viral Strains using Phylogenetic Analyses.

Abstract

ViralGeneClock is a Linux-based web application, where users can submit FASTA sequences of viral strains. The tool utilizes the Neighbor-Joining (NJ) algorithm to compute branch lengths and estimate the relative mutation rate for each gene across various strains. *ViralGeneClock* facilitates understanding evolutionary ties between the strains and identifies viral sites with rapid mutation as well as areas of high conservation. This tool integrates external tools such as *Prokka* and MUSCLE into a coherent pipeline - *Prokka* is employed for gene annotation and MUSCLE is used for running multiple sequence alignment (MSA). Furthermore, Biopython has been used for managing FASTA files, and Flask for creating a user-friendly web interface. Users can submit a FASTA sequence of different viral strains, reference genome strain name, and email address. The result is then emailed back to the user upon the completion of the analysis. Tests with sequences from SARS-CoV-2, HIV-1, and Influenza-A have demonstrated that the tool's outputs are consistent with the existing qualitative data from the scientific literature.

Introduction

In rapidly mutating viruses, phylogeny can illustrate evolutionary models and predict the potential emergence of new strains. The advancement of phylogenic models for analyzing viral evolution has led to the widespread adoption of these tools for vaccine and antiviral development [1]. While multiple tools utilize the sophisticated Maximum Likelihood (ML) algorithms for whole-genome phylogenetic studies in viruses, there remains a gap in available lightweight tools that directly facilitate the comparison of mutation rates and molecular clocks across the genes among different viral strains.

ViralGeneClock is primarily a Linux web application developed through the Ubuntu subsystem, where the users can deposit the whole genome sequence (WGS) of different viral strains. *ViralGeneClock* then examines the evolutionary relationship of the strains and quantitates the relative mutation rates of the genes. The rationale for the development of this tool are as follows:

1. *ViralGeneClock* identifies viral genes with a low mutation rate/high conservation. This could potentially offer vital insights for drug and vaccine development. Focusing on regions prone to rapid mutation for drug/vaccine might lead to challenges, as the virus could mutate beyond the scope of the treatment over time. Therefore, leveraging information on conserved regions, along with specific studies on the virus itself, could serve as a promising starting point for drug/vaccine development. The approach of targeting conserved regions is being employed in SARS-CoV-2 drug research; RNA-targeting molecules, which attack the conserved RNA structures and sequences, are being emphasized as potential drug candidates [2].

2. *ViralGeneClock* facilitates understanding the genetic distances and evolutionary ties between different viral strains.
3. *ViralGeneClock* also identifies viral genes with rapid mutation, indicating that the region is under strong selective pressure. Recognizing these areas prone to rapid mutation, researchers could potentially anticipate the emergence of new viral strains.

ViralGeneClock utilizes the Neighbor-joining (NJ) algorithm for phylogenetic analyses.

Neighbor-joining (NJ) is a bottom-up clustering method for estimating genetic distances and branch lengths, and creating phylogenetic trees [3]. While the neighbor-joining algorithm has largely been replaced by other algorithms with superior accuracy such as Maximum Likelihood and Maximum Parsimony, NJ remains the least computationally expensive algorithm among them [4]. This makes NJ appropriate for analyzing large data sets in a local device. Based on the predicted genetic distance of each strain with the reference strain for a gene, the relative mutation rate can be estimated by dividing the genetic distance by the branch length of the strain to the reference strain.

$$\text{Estimated Relative Mutation Rate for a Gene} = \frac{\text{Genetic distance to the Reference Strain for the Gene}}{\text{Branch length to the Reference Strain}}$$

While this estimate will not provide an absolute mutation rate for the gene, the ratio of mutation rates estimated for different genes can give insight into their relative mutation rates.

Tools & Materials

a) *Prokka*

ViralGeneClock leverages *Prokka* [5], a Linux-based rapid prokaryotic genome annotation tool, for viral annotation. *Prokka* uses an external feature prediction tool, *Prodigal*, to identify the

coordinates of protein-coding CDS. *Prodigal* is a non-supervised machine learning algorithm, which detects protein-coding regions by scanning for open reading frames (ORFs), and assigning scores to each ORF based on codon usage bias, GC frame plot, and length of the ORF [6]. *Prokka* then compares the gene code at a protein sequence level with databases of known sequences to annotate the predicted gene.

b) MUSCLE

ViralGeneClock uses MUSCLE (Multiple Sequence Comparison by Log-Expectation) for multiple sequence alignment of whole genome sequences of viral strains, as well as on each individually annotated gene. MUSCLE takes homologous sequences as an input, and employs a progressive alignment algorithm [7]. In the progressive alignment algorithm, the sequences are initially pairwise aligned based on similarity scores, and then these pairwise alignments are progressively aligned into larger alignments.

c) Python3 Dependencies: Biopython, Flask & Flask-Mail

ViralGeneClock uses Biopython to manage, clean, and organize the FASTA files [8]. The *Phylo* library within Biopython is also leveraged to produce a phylogenetic tree diagram for viral strains (Appendix A). The command line interface of *ViralGeneClock* has been wrapped with Flask, a lightweight Python web application framework [9], to provide a user-friendly web application interface. Additionally, Flask-Mail, an extension of the Flask framework [10], automatically emails users with results once the phylogenetic analysis is complete.

d) HTML, CSS, and JavaScript

HTML and CSS have been employed to design the front end of the *ViralGeneClock* web application. JavaScript's AJAX Object has also been utilized to retrieve live updates from the

command line and display them on the webpage without requiring the user to reload the page [11].

e) Input FASTA Files

The input file (whole genome viral sequences) for *ViralGeneClock* can be obtained from [NCBI Virus](#). A sample input *sample_input.fasta* (Appendix B) containing sequences for 10 SARS-CoV-2 strains was retrieved via NCBI's e-utilities using *sarsCov2DataExtract.py* (Appendix C).

Methods and Algorithms

a) Full Sequence Analysis

- Genome Annotation and Multiple Sequence Alignment:

ViralGeneClock first annotates the whole genome sequence of the reference strain using *Prokka*. Then, MUSCLE is leveraged to run a progressive multiple-sequence alignment of the whole genome sequence of all the submitted viral strains. This process is run by the Python script *prokkaMuscleFullSequence.py* (Appendix D), and the resulting outputs are a .tsv file with genome annotation for the reference strain and a MUSCLE-produced alignment file.

- Neighbor-Joining (NJ) Algorithm:

The alignment file is then used for running the neighbor-joining algorithm, which clusters the strains according to their genetic distances (Appendix E). This neighbor-joining pipeline also involves bootstrapping to create multiple pseudo-datasets (*bootstrap.py*, Appendix E). This increases the robustness of the model and helps avoid biases from sampling variation. This algorithm produces a genetic distance matrix and computes branch lengths for all the strains.

The script *tree_maker.py* utilizes the branch lengths calculated by the NJ algorithm to produce a phylogenetic tree image file.

b) Gene Analysis

- Genome Annotation, Gene Grouping & Multiple Sequence Alignment for Each Gene:

Initially, *ViralGeneClock* runs *Prokka*'s annotation for each strain. Following this, the sequences of each gene from the strains are sorted into individual files with the *prokkaMuscle.py* (Appendix F) script. For each of these gene-specific files, MUSCLE runs a progressive MSA to produce alignment files.

- Neighbor-Joining (NJ) Algorithm:

For each gene-specific alignment file, scripts running the NJ algorithm calculate the genetic distance matrix (Appendix E).

- Estimation of Relative Mutation Rate:

From the genetic distance matrix for each gene, the genetic distance of each strain to the reference strain is determined for that specific gene. The mutation rate for a gene is then calculated by dividing its genetic distance (obtained from the gene's genetic distance matrix) by the branch length of the strain to the reference strain (derived previously from the full sequence analysis). The mutation rates for each gene are then averaged across all strains to determine the relative mutation rate for that gene (Appendix G). This method assumes that the mutation rate is uniform across all strains for a given gene.

c) Script Consolidation and Web Application Operations

Finally, the Python scripts have been consolidated to automate the entire process;

main_GeneAnalysis.py automates the gene analysis section and *main_fullSequence.py* automates the full sequence analysis section (Appendix H).

The command line interface of *ViralGeneClock* has been integrated with Flask to provide a web application interface. The script *app.py* (Appendix I) displays the front-end *index.html* (Appendix J) when run, which accepts the FASTA sequence, reference genome strain name, and email address from the user. After submission, the flask script calls on *running.html* (Appendix J), which incorporates JavaScript's AJAX Object, to retrieve live updates from the terminal every 10 seconds. After running the full sequence and gene analysis, *app.py* (Appendix I) emails the zipped results to the user using Python's Flask-Mail library.

Tool Workup

a) Input

ViralGeneClock takes the WGS of viral strains in FASTA format as an input. The FASTA header of each strain should be the name of the strain. Some sample inputs are in the mkayasth/viralGeneClock--codespaces [GitHub repository](#):

- Sample_input.fasta: 10 SARS-CoV-2 viral strains.
- HIV-samples.txt: 7 HIV-1 viral strains.
- InfluenzaVirus-samples.txt: 4 Influenza A viral strains.

b) GitHub Codespace/Linux Environment Setup

ViralGeneClock uses some Linux-based dependencies, which need to be installed before running the tool. The instructions for prerequisite installations in a Linux system are in the [Readme.md](#) file in the [mkayasth/viralGeneClock](#) GitHub repository.

GitHub Codespaces can be used in other operating systems to emulate the Linux environment. Codespaces are development environments from GitHub that are hosted in the cloud. The instructions for installing the dependencies for *ViralGeneClock* in the virtual environment are in the [Readme.md](#) file in the [mkayasth/viralGeneClock--codespaces](#) repository.

c) Web Application Operations

- After installing the prerequisites of *ViralGeneClock*, the web app can be accessed by running `app.py` (Appendix I).

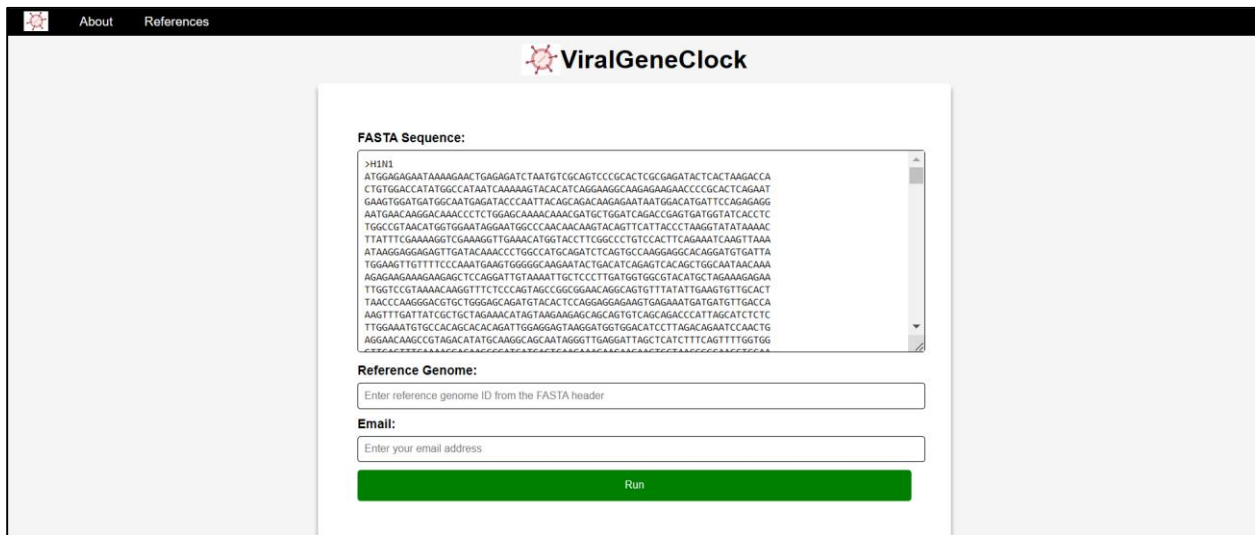
The screenshot shows the front-end of the ViralGeneClock web application. At the top, there is a navigation bar with 'About' and 'References' links. The main header features the 'ViralGeneClock' logo, which is a red virus icon. Below the header, the interface is divided into two main sections. The left section, titled 'FASTA Sequence:', contains a large text area with a scroll bar, displaying a long DNA sequence starting with '>H1N1'. The right section, titled 'Reference Genome:', has a text input field with the placeholder 'Enter reference genome ID from the FASTA header'. Below this, there is an 'Email:' label and another text input field with the placeholder 'Enter your email address'. At the bottom of the right section, there is a prominent green button labeled 'Run'.

Figure 1: Front-end for the ViralGeneClock Web Application.

- After submitting the FASTA sequence, the reference genome, and the email address, *ViralGeneClock* captures real-time updates from the terminal every 10 seconds to display on the screen.

Task Status: Running

```
[14:30:53] This is prokka 1.14.6
[14:30:53] Written by Torsten Seemann <torsten.seemann@gmail.com>
[14:30:53] Homepage is https://github.com/tseemann/prokka
[14:30:53] Local time is Mon May 6 14:30:53 2024
[14:30:53] You are codespace
[14:30:53] Operating system is linux
[14:30:53] You have BioPerl 1.7.8
Argument "1.7.8" isn't numeric in numeric lt (<) at /opt/conda/bin/prokka line 259.
[14:30:53] System has 2 cores.
[14:30:53] Option --cpu asked for 8 cores, but system only has 2
[14:30:53] Will use maximum of 2 cores.
[14:30:53] Annotating as >>> Viruses <<<
[14:30:53] Generating locus_tag from 'fullSequence-output/H1N1_sequence.fasta' contents.
[14:30:53] Setting --locustag OIPHMFG from MD5 82916f70fa96e8c7870e671d5ceb29c3
[14:30:53] Re-using existing --outdir fullSequence-output
[14:30:53] Using filename prefix: H1N1.XXX
[14:30:53] Setting HAMMER_NCPU=1
[14:30:53] Writing log to: fullSequence-output/H1N1.log
[14:30:53] Command: /opt/conda/bin/prokka --force fullSequence-output/H1N1_sequence.fasta --outdir fullSequence-output --prefix H1N1 --
kingdom Viruses
```

Figure 2: Real-time update displayed on the screen after submission.

- Once the process has been successfully completed, a success page is displayed. The outputs are compressed into a zip file and emailed back to the user.

Your request has been processed successfully! **outputs.zip** has been emailed.

Output files:

- FOLDER fullSequence-output: gene annotation from Prokka, Full sequence MSA, Genetic distance matrix between the strains, Phylogenetic tree.
- FOLDER geneAnalysis-output: MSA across the viral strains for each gene.
- FOLDER avg_mutation_rate_final: relative mutation rate for each gene across the viral strains.

Figure 3: Web App Notification indicating results have been successfully delivered to the user.

- In case of any error, the details of the error are captured from the terminal and displayed on the screen. If the provided FASTA sequence has a structure error or the reference strain provided is not found in the header of any FASTA sequence, the user is notified of the problem.

Failed to process your request. Error with the FASTA sequence / reference genome not found in any FASTA header.

```
[14:54:40] This is prokka 1.14.6
[14:54:40] Written by Torsten Seemann <torsten.seemann@gmail.com>
[14:54:40] Homepage is https://github.com/tseemann/prokka
[14:54:40] Local time is Mon May 6 14:54:40 2024
[14:54:40] You are codespace
[14:54:40] Operating system is linux
[14:54:40] You have BioPerl 1.7.8
Argument "1.7.8" isn't numeric in numeric lt (<) at /opt/conda/bin/prokka line 259.
[14:54:40] System has 2 cores.
[14:54:40] Option --cpu asked for 8 cores, but system only has 2
[14:54:40] Will use maximum of 2 cores.
[14:54:40] Annotating as >>> Viruses <<<
[14:54:40] 'fullSequence-output/H1N1_sequence.fasta' is not a readable non-empty FASTA file
Traceback (most recent call last):
  File "/workspaces/viralGeneClock--codespaces/fullSequenceAnalysis/prokkaMuscleFullSequence.py", line 126, in <module>
    main()
```

Figure 4: Notification indicating an error when the FASTA sequence is entered in the wrong format.

d) Output

The output folder emailed to the user consists of three sub-folders: avg_mutation_rate_final, geneAnalysis-output, and fullSequence-output.

Folder avg_mutation_rate_final consists of average_mutation_rates.txt, which provides the estimated relative mutation rate for each gene annotated by *Prokka* across the strains. The folder geneAnalysis-output consists of MUSCLE-produced multiple sequence alignment files (.ffn format) for each gene across the viral strains. The folder fullSequence-output consists of genome annotation from *Prokka* in a .tsv format, a whole genome multiple sequence alignment file (.ffn format), a genetic distance matrix between the strains, tree.txt (depicting branch lengths across the strains in a Newick format), and a phylogenetic tree image file produced using the tree.txt data.

Results

Sample #1: Sample_input.fasta (10 SARS-CoV-2 viral strains).

locus_tag	ftype	length_bp	gene	EC_number	COG	product
NAFDKFMI_00001	CDS	13218	1a			Replicase polyprotein 1a
NAFDKFMI_00002	CDS	7788	rep			Replicase polyprotein 1ab
NAFDKFMI_00003	CDS	3822	S			Spike glycoprotein
NAFDKFMI_00004	CDS	828	3a			Protein 3a
NAFDKFMI_00005	CDS	669	M			Membrane protein
NAFDKFMI_00006	CDS	186				hypothetical protein
NAFDKFMI_00007	CDS	366	7a			Protein 7a
NAFDKFMI_00008	CDS	366				hypothetical protein
NAFDKFMI_00009	CDS	1260	N			Nucleoprotein

Figure 5: *Prokka* genome annotation for SARS-CoV-2 virus.

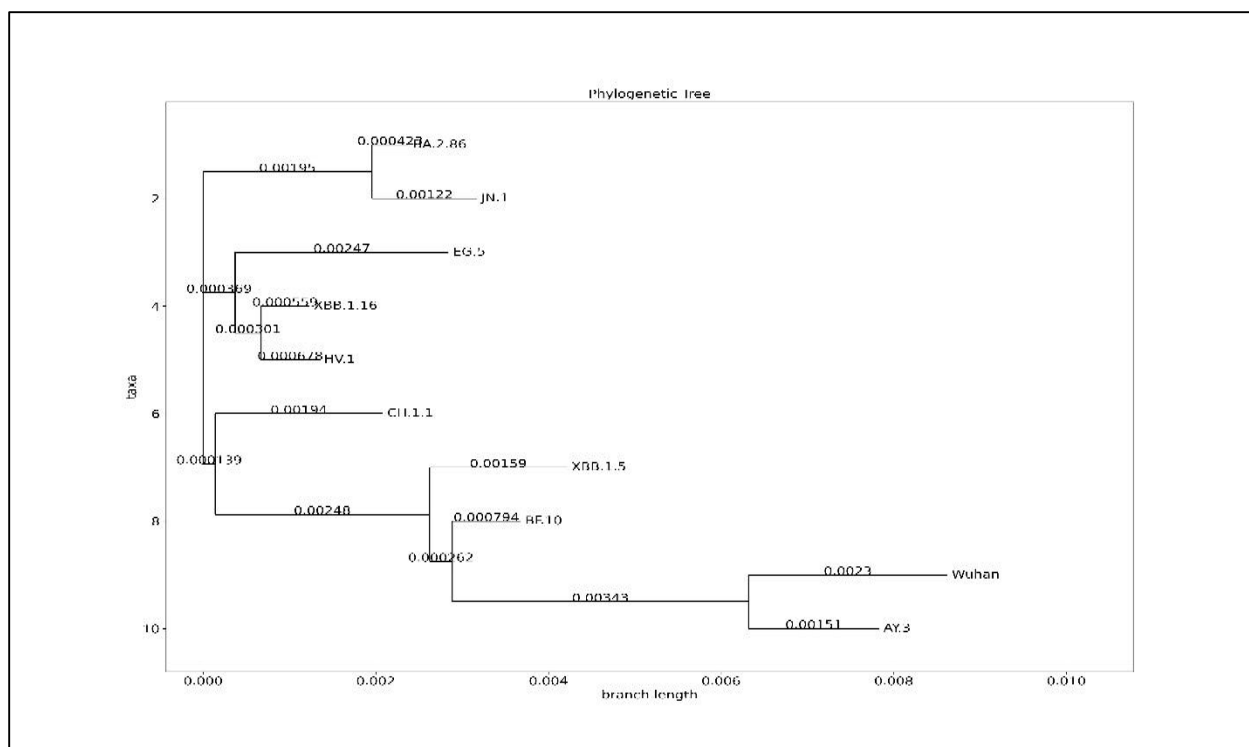


Figure 6: Phylogenetic tree of 10 SARS-CoV-2 strains produced by *ViralGeneClock*.

Average mutation rate for Membraneprotein: 0.650966790336159
Average mutation rate for Spikeglycoprotein: 1.6416065098551205
Average mutation rate for polyprotein1a: 0.24757839184013594
Average mutation rate for Protein7a: 0.30272380462484016
Average mutation rate for EGFOJODP_00009Nucleoprotein: 1.316917919005337
Average mutation rate for Protein3a: 0.383259674475547
Average mutation rate for polyprotein1ab: 0.17520089761368174
Figure 7: Estimated relative mutation rate for all annotated genes in SARS-CoV-2 (generated from 10 viral strains).
Highest relative mutation rate: Spike glycoprotein; Lowest relative mutation rate: Poly protein 1ab.

Sample #2: HIV-samples.txt (7 HIV-1 viral strains).

locus_tag	ftype	length_bp	gene	EC_number	COG	product
EHFMEDCL_00001	CDS	1506	gag		Gag	polyprotein
EHFMEDCL_00002	CDS	2709	gag-pol		Gag-Pol	polyprotein
EHFMEDCL_00003	CDS	579	vif		Virion	infectivity factor
EHFMEDCL_00004	CDS	291	vpr		Protein	Vpr
EHFMEDCL_00005	CDS	246	vpu		Protein	Vpu
EHFMEDCL_00006	CDS	2502	env		Envelope	glycoprotein gp160
EHFMEDCL_00007	CDS	654	nef		Protein	Nef

Figure 8: *Prokka* genome annotation for HIV-1 virus.

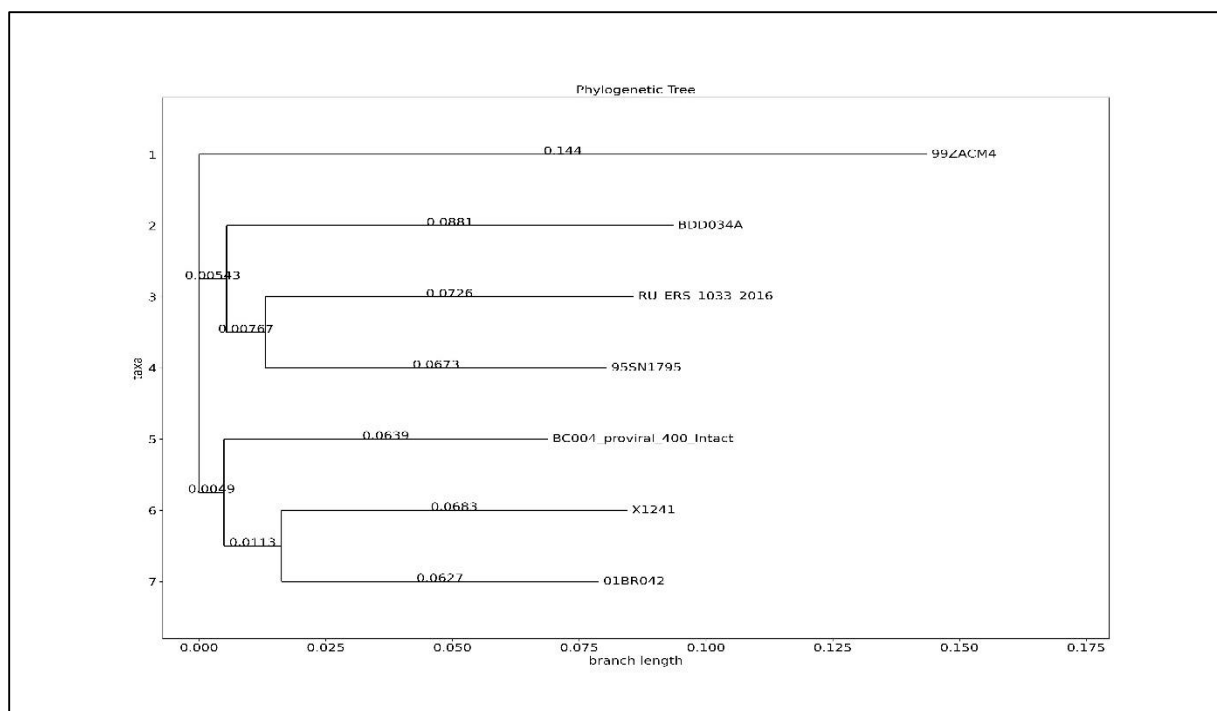


Figure 9: Phylogenetic tree of 7 HIV-1 strains produced by *ViralGeneClock*.

Average mutation rate for ProteinNef: 0.9063598491278966
Average mutation rate for infectivityfactor: 0.6173213233104081
Average mutation rate for glycoproteingp160: 0.8955625774549426
Average mutation rate for Gagpolyprotein: 0.6283936466068032
Average mutation rate for ProteinVpr: 0.6537817348916457
Average mutation rate for ProteinVpu: 0.993105920040645
Average mutation rate for Gag-Polpolyprotein: 0.4516290405959998
Figure 10: Estimated relative mutation rate for all annotated genes in HIV-1 virus (generated from 7 viral strains).
Highest relative mutation rate: ProteinVpu, ProteinNef, Glycoproteingp160;
Lowest relative mutation rate: Gag-Pol-poly protein.

Sample #3: InfluenzaVirus-samples.txt (4 Influenza A viral strains).

locus tag	ftype	length bp	gene	EC number	COG product
OIPHMFNG_00001	CDS	2280	PB2		Polymerase basic protein 2
OIPHMFNG_00002	CDS	2274	PB1	2.7.7.48	RNA-directed RNA polymerase catalytic subunit
OIPHMFNG_00003	CDS	2118	PA	3.1.-.-	Polymerase acidic protein
OIPHMFNG_00004	CDS	1701	HA		Hemagglutinin
OIPHMFNG_00005	CDS	1497	NP		Nucleoprotein
OIPHMFNG_00006	CDS	1368	NA	3.2.1.18	Neuraminidase
OIPHMFNG_00007	CDS	759 M			Matrix protein 1
OIPHMFNG_00008	CDS	660 NS			Non-structural protein 1

Figure 11: Prokka genome annotation for Influenza A virus.

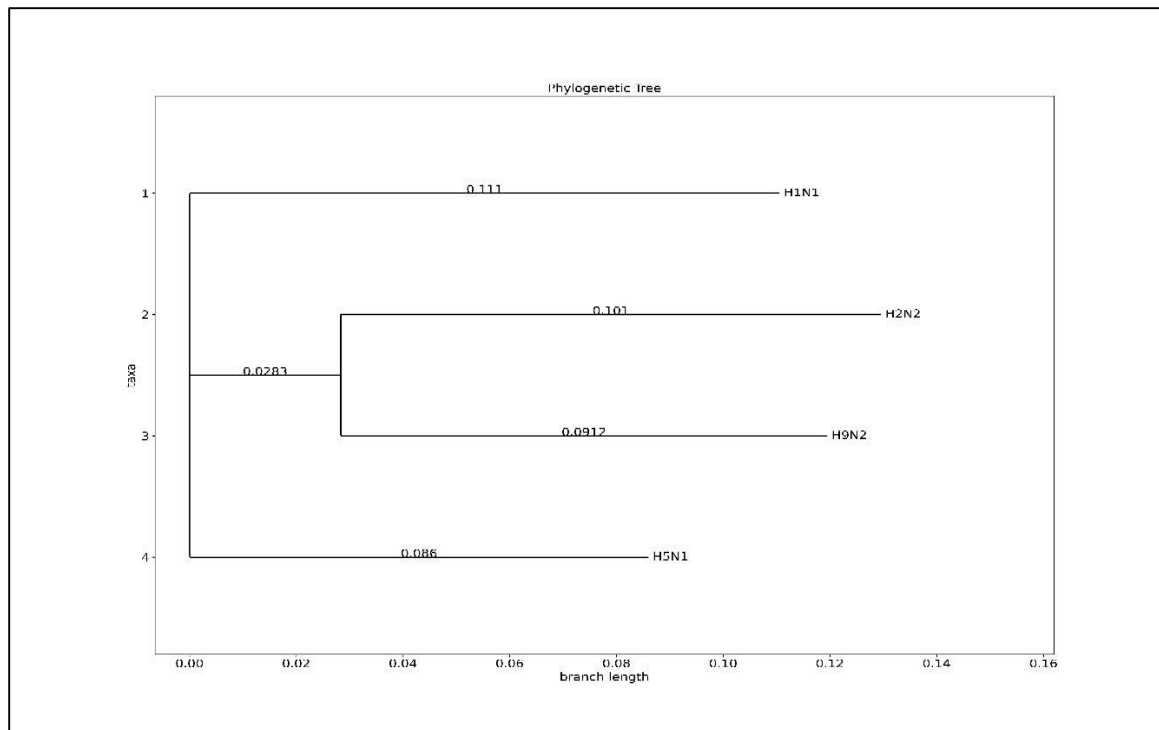


Figure 12: Phylogenetic tree of 4 Influenza A strains produced by *ViralGeneClock*.

Average mutation rate for NMCHHEOD_00005Nucleoprotein: 0.7519412408169522
 Average mutation rate for protein1: 1.2006538657037895
 Average mutation rate for acidicprotein: 0.6421659772966403
 Average mutation rate for catalyticsubunit: 0.5975132953040165
 Average mutation rate for protein1: 0.43340307740013245
 Average mutation rate for GBOAADJJ_00006Neuraminidase: 1.7229239933563172
 Average mutation rate for NMCHHEOD_00004Hemagglutinin: 1.7463257465660276
 Average mutation rate for protein2: 0.7141895603686917

Figure 13: Estimated relative mutation rate for all annotated genes in Influenza A (generated from 4 viral strains).

Highest relative mutation rate: Hemagglutinin, Neuraminidase;

Lowest relative mutation rate: protein1.

Discussion

For the 10 SARS-CoV-2 viral strains, Spike glycoprotein (S) was obtained to have the highest relative mutation rate of 1.642 from *ViralGeneClock*, followed by Nucleoprotein (N) with a relative mutation rate of 1.317. Past studies have shown that S and N genes both present a

high mutational range [12]. The spike protein's receptor-binding domain (RBD) can alter its binding affinity with the ACE2 receptor, which increases the virus's ability to evade the immune responses [12]. Furthermore, the N protein is responsible for packaging the viral RNA and influences the detection of the virus by the immune system. Studies have also reported that R203K/G204R modifications in the N protein are associated with high viral replication, infectivity, and transmissibility [12]. So, the high mutation rate in the S and N genes is likely driven by the high evolutionary pressure.

In the 7 HIV-1 viral strains, *ViralGeneClock* found the highest relative mutation rates in Protein Vpu (0.993), Protein Nef (0.906), and Glycoprotein gp160 (0.896). According to other studies, the envelope genes (gp160, a precursor to form surface glycoprotein gp120 and gp41) have been observed to have the highest mutation rates [13]. In Influenza A strains, *ViralGeneClock* found the highest relative mutation rates in Hemagglutinin (HA) and Neuraminidase (NA) with a value of 1.746 and 1.723 respectively. This result is also supported by other studies from the scientific literature; surface protein genes HA and NA have been found to evolve more rapidly than internal protein genes [14]. HA, in particular, helps in attaching to and entering the cells in the respiratory tract, and regular mutations in the HA gene help the virus evade the immune response.

Due to a paucity of literature on quantitative mutation rates across different genes in viral strains, verifying the accuracy of *ViralGeneClock*'s quantitative relative mutation rates is difficult. However, *ViralGeneClock*'s findings align with the qualitative information in the literature, as observed in the examples of SARS-CoV-2, HIV-1, and Influenza 1. *ViralGeneClock* is effective in identifying the evolutionary relationship between different viral strains and can be utilized for pinpointing viral regions with varying levels of conservation. The

quantitative accuracy, however, may be limited by the use of the Neighbor-Joining algorithm over other computationally heavy, modern phylogenetic algorithms. *ViralGeneClock* also assumes that the mutation rate for a gene is constant across different viral strains, which is not the case in nature.

Conclusion

ViralGeneClock effectively annotates the input FASTA sequences and estimates the relative mutation rates of each gene across the strains. While the quantitative accuracy of *ViralGeneClock* remains to be validated, the qualitative results are promising. Being a relatively lightweight tool, *ViralGeneClock* enables the identification of highly conserved and variable sites within a genome on a local server. This data can be leveraged as a starting point for drug/vaccine research, exploring the evolutionary relationship between viral strains, and predicting the potential future strains.

References

1. Cardona-Ospina, J. A., Rojas-Gallardo, D. M., Garzón-Castaño, S. C., Jiménez-Posada, E. V., & Rodríguez-Morales, A. J. (2021). Phylodynamic analysis in the understanding of the current COVID-19 pandemic and its utility in vaccine and antiviral design and assessment. *Human Vaccines & Immunotherapeutics*, 17(8), 2437–2444. <https://doi.org/10.1080/21645515.2021.1880254>
2. Hegde, S., Tang, Z., Zhao, J., & Wang, J. (2021). Inhibition of SARS-CoV-2 by targeting conserved viral RNA structures and sequences. *Frontiers in Chemistry*, 9. <https://doi.org/10.3389/fchem.2021.802766>
3. Saitou, N., & Nei, M. (1987). The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4), 406–425. <https://doi.org/10.1093/oxfordjournals.molbev.a040454>
4. Kuhner, M. K., & Felsenstein, J. (1994). A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. *Molecular Biology and Evolution*, 11(3), 459–468. <https://doi.org/10.1093/oxfordjournals.molbev.a040126>
5. Seemann, T. (2014). Prokka: rapid prokaryotic genome annotation. *Bioinformatics* (Oxford, England), 30(14), 2068–2069. <https://doi.org/10.1093/bioinformatics/btu153>
6. Hyatt, D., Chen, G.-L., LoCascio, P. F., Land, M. L., Larimer, F. W., & Hauser, L. J. (2010). Prodigal: prokaryotic gene recognition and translation initiation site identification. *BMC Bioinformatics*, 11(1). <https://doi.org/10.1186/1471-2105-11-119>
7. Edgar, R. C. (2004). MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5), 1792–1797. <https://doi.org/10.1093/nar/gkh340>
8. Cock, P. J. A., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B., & de Hoon, M. J. L. (2009). Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* (Oxford, England), 25(11), 1422–1423. <https://doi.org/10.1093/bioinformatics/btp163>
9. Flask. Readthedocs.org. Retrieved April 6, 2024, from <https://readthedocs.org/projects/flask/>
10. Flask-mail — Flask-Mail 0.9.1 documentation. Pythonhosted.org. Retrieved May 1, 2024, from <https://pythonhosted.org/Flask-Mail/>
11. AJAX introduction. W3schools.com. Retrieved May 1, 2024, from https://www.w3schools.com/js/js_ajax_intro.asp
12. Hirabara, S. M., Serdan, T. D. A., Gorjao, R., Masi, L. N., Pithon-Curi, T. C., Covas, D. T., Curi, R., & Durigon, E. L. (2022). SARS-COV-2 variants: Differences and potential of immune evasion. *Frontiers in Cellular and Infection Microbiology*, 11. <https://doi.org/10.3389/fcimb.2021.781429>
13. Cuevas, J. M., Geller, R., Garijo, R., López-Aldeguer, J., & Sanjuán, R. (2015). Extremely high mutation rate of HIV-1 in vivo. *PLoS Biology*, 13(9), e1002251. <https://doi.org/10.1371/journal.pbio.1002251>
14. Webster, R. G., Bean, W. J., Gorman, O. T., Chambers, T. M., & Kawaoka, Y. (1992). Evolution and ecology of influenza A viruses. *Microbiological Reviews*, 56(1), 152–179. <https://doi.org/10.1128/mr.56.1.152-179.1992>