SARS2EVO

The site includes all custom scripts and files used in the study of “Underlying driving forces of the SARS-CoV-2 evolution: immune evasion and ACE2 binding affinity”.

The code includes python scripts, C++ scripts, and C# scripts. Both Linux and Windows platforms are required. We recommend running the scripts on a Windows working station with 128Gb RAM and enabling the Windows subsystem Linux (WSL) system to avoid intermediate file transfer which can be very time-consuming. In this documentation, unless otherwise noted, all steps are run under the Windows system.

Most of the scripts are written in Microsoft C# language based on the .net framework, the original code and project files are provided. To run the scripts, Microsoft Visual Studio is needed. The directory of the input and output should be named in the script before hitting the Compile and run button.

1. For data collection,

a total number of 6,484,070 high-quality open-access SARS-CoV-2 sequences and corresponding metadata were downloaded from the UShER website on 23 November 2022 as the input. The following files were downloaded and unzipped:

public-latest.all.masked.pb.gz

public-latest.metadata.tsv.gz

public-latest.all.masked.vcf.gz

(http://hgdownload.soe.ucsc.edu/goldenPath/wuhCor1/UShER_SARS-CoV-2/)

2. Calculate mutation rate (Figure 1)

2.1 Obtain mutation and metadata for each sequence

2.1.1Transfer the unzipped VCF file to *mut5col* format using Vcf2mut5col (Linux).

“Vcf2mut5col.out *public-latest.all.masked.vcf* [output directory]”*

*This will generate a *.mut5col* file.

2.1.2Transfer the *.mut5col* file to *sample_mutlist.tsv* using Mutresult2Sample_mutlistLINUX (Linux).

<u>"Mut5col2Sample_mutlistLINUX.out [directory]"*</u>

*Both input and output will be in this directory. The *sample_mutlist.tsv* is a txt file that stores mutation information of each sequence in the format of [SequenceName]:Mutation1,Mutation2,...,MutationN.

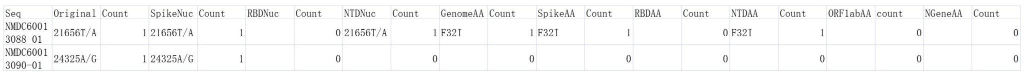
2.2 Random selection of sequences

Random selection of 200 sequences each month using SampleMutlistFilterAndRandomPick. The input are the *sample_mutlist.tsv* file and the *public-latest.metadata.tsv* file. This will generate the *sampled_sample_mutlist.tsv* and the *sampled_metadata.tsv* file.

2.3 Sequence mutation annotation

Before plotting, the *sampled_sample_mutlist.tsv file* was further annotated, transfer each nucleotide mutation to amino acid mutation, generating a result table file *sampled_sample_mutlist.transfer.tsv* using SampleMutlistTransfer.

*The *sampled_sample_mutlist.transfer.tsv* can be directly combined with the *sampled_metadata.tsv* file in the Excel for they share the same sequence order. The output file will be the following format.



Seq	Original	Count	SpikeNuc	Count	RBDNuc	Count	NTDNuc	Count	GenomeAA	Count	SpikeAA	Count	RBDAA	Count	NTDAA	Count	ORF1abAA	count	NGeneAA	Count
NMDC6001 3088-01	21656T/A	1	21656T/A	1		0	21656T/A	1	F32I	1	F32I	1		0	F32I	1		0		0
NMDC6001 3090-01	24325A/G	1	24325A/G	1		0		0		0		0		0		0		0		0

2.4 Mutation rate calculate

After combining, the transferred table was then plotted using the python script segments_fit.py (for the automatic piecewise linear regression) and the R ggplot function (for simple linear regression).

3. Find all mutation events

The downloaded protobuf file *public-latest.all.masked.pb.gz* was first transferred to a readable JSON file using matUtils (Linux) from the UshER toolkit. (https://usher-wiki.readthedocs.io/en/latest/)

"matUtils extract -i ./public-latest.all.masked.pb.gz -d [*YOUR DIRECTORY*] -j [*MAT1128*].json"

*This step should be run on Linux and can require memory usage bigger than 65Gb.

Then we search for all mutation events from the MAT tree using Json2MutationEvent_C#. It will generate a *.json.mutevent* file. The file contains all filtered sequences (see methods section) and their additional mutations relative to ancestors.

*The input files are the *public-latest.metadata.tsv* and the *.json* file. The output will be in the following format.

Sample	AccessionID	OriginalMut	NewMut	Lineage	CollectionDate	Location	SameSeqNumber	NextstrainClade
CHN/YN-0306-466/2020|MT396241.1|2020-03-06	CHN/YN-0306-466/2020|MT396241.1|2020-03-06	G15910T	G15910T	B	2020/3/6	China	0	19A
England/SHEF-BFCA2/2020|2020-03-01	England/SHEF-BFCA2/2020|2020-03-01	C24372T, G29779C	C24372T	B	2020/3/1	England	0	19A

4. Calculate mutation distribution on the genome (Figure 1)

Mutation distribution on the genome and collection date was extracted from the obtained *.mutevent* file using GenomeMutationDistribution. There will be several output files:

File	Description
MutationRate.txt	The distribution of mutations across different genomic regions over time.
NSMutRate.txt	The distribution of Nonsynonymous mutations across different genomic regions over time.
SMutRate.txt	The distribution of Synonymous mutations across different genomic regions over time.
MutationCount.txt	Event counts of each mutation.
Spike50.txt	The proportion of mutations that accounted for 50% of all mutations events (P50) at different time points.

5. Calculate the mutation incidence in different SARS-CoV-2 lineages (Figure 2)

The mutation incidence of the most frequent mutations in different SARS-CoV-2 lineages was calculated using FindCladeSpecificMutation which can automatic generate several heatmap tables for each gene.

*The input file is the .json.mutevent and the output files are:

File	Description
CladeRBDMutRate.txt	Mutations under each Lineage and their incidence.
CladeHeatmap.txt	The incidence heatmap of top frequent mutations.

The heatmap was then plotted with the R pheatmap package. The principal component analysis (PCA) plot was plotted with the R fviz package.

6. Calculate the Silhouette Coefficient (Figure 2)

After obtaining the top most frequent mutations table, we calculated the Silhouette Coefficient of each gene and clade using CalSilhouetteCoefficient.

*The input files are the *CladeHeatmap.txt* file of each gene.

7. Calculate the distribution of escape mutations (Figure 3)

7.1 Calculate the escape score

The antibody spectrum, neutralizing activity, antibody epitope group, and raw mutation escape score were obtained from previous studies (see methods). We combined the raw mutation escape score with antibody-neutralizing activity using CalMutEscapeScore.

*The input DMS and output files are the following:

File	Description
use_abs_res.csv	Raw escape scores.
NeutralWTBA125_Cross.txt	Antibody neutralization data.
single_mut_effects.csv	ACE2 binding and RBD expression data.
EscapeScore_PKU.single.txt	The processed escape score.
EscapeScore_PKU.12.txt	The processed escape score of the 12 epitopes.

7.2 Assign escape mutations

The mutation that significantly reduced the affinity to any of the 12 antibody types was defined as an immune escape mutation. The escape mutations were assigned based on the escape score in the file *EscapeScore_PKU.12.txt* using EachCladeEscapeScoreTo12Epitope. The output escape mutation distribution heatmap file *CladeRBDMut12Group.txt* was also calculated using this script.

7.3 Calculate the antibody pressure

The immune pressure exerted on a particular epitope region is calculated by summing the neutralizing activities of all antibodies that belong to this epitope group using Cal12EpiPressureDistribution.

*The input file is the antibody neutralization table *NeutralWTBA125_Cross.txt* and the output files are the *AntibodySpectrum12_Count.txt* and the *AntibodySpectrum12_logIC50.txt* which represent the antibody number and pressure act on the 12 epitopes.

7.4 The gene set enrichment analysis (GSEA)

The GSEA analysis was also performed using the script FindCladeSpecificMutation, as one of its function.

*It will generate 2 files as output:

File	Description
Mutation_ES_Curve.txt	The GSEA plot curve data.
Mutation_ES_Pvalue.txt	The p-value after 50000 randomization repeating.

8. Calculate the evolution trajectory (Figure 4)

8.1 Construct a tree of Lineages

To calculate the evolution trajectory, it is necessary to restore the evolutionary relationship of all Lineages. We obtained the Lineage list from the Pango website and refined the relationships into *lineageRelations.tsv* using LineageRelations.

(https://github.com/cov-lineages/lineages-website)

*input: *lineages.yml*

Output: *lineageRelations.tsv*

8.2 Summarize genome mutations and metadata for each Lineage

We first combined the *sample_mutlist.tsv* file with the *metadata.tsv* file using

SeqFindMeta (Linux). Then, we counted the sampling time and genome mutation of each Lineage sequence using LineageMutationMetadata, generating the *Lineage.Date.AAmut* file:

Lineage	CollectionDate5P	CollectionDate25P	Mut70P	CountryNumber	SeqNumber
B	20200130	20200304		74	2521
BF. 5	20220629	20220719	G339D, S371F, S373P, S375F, T376A, D405N, R408S, K417N, N440K, L452R, S477N, T478K, E484A, F486V, Q498R, N501Y, Y505H	63	22766
BA. 5. 2	20220621	20220715	G339D, S371F, S373P, S375F, T376A, D405N, R408S, K417N, L452R, S477N, T478K, E484A, F486V, Q498R, N501Y, Y505H, N440K	93	93996

8.3 Calculate the evolution trajectory

Finally, we combined the above result with the escape score and ACE2 binding data, using LineageTrajectory, generating a table *LineagePlotData.txt* that can be directly used for R ggplot.

Lineage	VOC	Date5P	CountryNumber	SeqNumber	DMS_Infect	DMS_Escape	BranchGroup	Midpoint	NewMut	BackMut	ΔESC	ΔACE
A. 2. 5	Other	20210102	25	969	0. 23469	22. 41524934	A. 2. 5	Lineage	452R	NA	22. 41524934	0. 23469
A. 2	Other	20200303	32	410	0	0	A. 2. 5	Midpoint		NA	0	0
A. 2. 5. 3	Other	20210506	4	30	0. 97225	26. 85109422	A. 2. 5. 3	Lineage	477N	NA	4. 435844888	0. 73756
A. 2. 5	Other	20210102	25	969	0. 23469	22. 41524934	A. 2. 5. 3	Lineage	452R	NA	22. 41524934	0. 23469

9. Multivariate linear regression (Figure 4)

9.1 Calculation of the RoHo

The RoHo value of each mutation event was calculated using the matUtils (Linux) from the UshER toolkit.

```
“matUtils summary -i ./public-latest.all.masked.pb.gz -E RoHo.tsv”
```

And then extract the corresponding data of the target clade into *MutationRoHo.txt* using CalMutRoHo.

9.2 Multivariate linear regression

The mutation incidence, RoHo, and corresponding escape score and ACE2 binding data were combined manually using Microsoft Office Excel. The multivariate linear regression was conducted by the R lm function.

```
“lm.sol = lm(EventFraction~ scale(DMSACE) + scale(DMSESC), data = RoHo_WT)”
“summary(lm.sol)”
```