0. Install BiocManager Biostrings

- Install Biostrings

```
if (!require("BiocManager", quietly = TRUE))
    install.packages("BiocManager")

BiocManager::install("Biostrings")
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

'getOption("repos")' replaces Bioconductor standard repositories, see
'help("repositories", package = "BiocManager")' for details.
Replacement repositories:
    CRAN: https://cran.rstudio.com

Bioconductor version 3.20 (BiocManager 1.30.25), R 4.4.2 (2024-10-31)

Installing package(s) 'BiocVersion', 'Biostrings'

also installing the dependencies ‘zlibbioc’, ‘UCSC.utils’, ‘GenomeInfoDbData’, ‘BiocGeneri

Old packages: 'bit', 'cpp11'
```

1. Upload the files to Colab

- Note: the uploaded files will be deleted when the runtime is finished

2. Load the installed package Biostrings

```
library(Biostrings)
```

```
Loading required package: BiocGenerics

Attaching package: ‘BiocGenerics’

The following objects are masked from ‘package:stats’:

    IQR, mad, sd, var, xtabs

The following objects are masked from ‘package:base’:

    anyDuplicated, aperm, append, as.data.frame, basename, cbind,
    colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
    get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
    match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
    Position, rank, rbind, Reduce, rownames, sapply, saveRDS, setdiff,
    table, tapply, union, unique, unsplit, which.max, which.min

Loading required package: S4Vectors
```

```
Loading required package: stats4

Attaching package: ‘S4Vectors’

The following object is masked from ‘package:utils’:

    findMatches

The following objects are masked from ‘package:base’:

    expand.grid, I, unname

Loading required package: IRanges

Loading required package: XVector

Loading required package: GenomeInfoDb

Attaching package: ‘Biostrings’

The following object is masked from ‘package:base’:

    strsplit
```

3. Read the fasta file containing the selected sequences

```
seqs <- readDNAStringSet("/content/selected_seqs.fasta") # replace the path with the path t
```

4. DATA CLEANING

1. Load the provided SARS-CoV-2 FASTA file `selected_seqs.fasta` (20 sequences, each 29,903 bp) into the Colab notebook.
2. Keep sequences with coverage over 85% (counting only A, C, T, and G bases).

Using a built-in function to calculate the proportion of A, C, T, G in the sequences

```
seqs_nt_count <- alphabetFrequency(seqs)
print(seqs_nt_count)
```

```
          A    C    G    T M R W S Y K V H D B     N  - + .
 [1,] 8934 5452 5838 9599 0 0 0 0 0 0 0 0 0 0    27 53 0 0
 [2,] 8883 5434 5827 9574 0 0 0 0 0 0 0 0 0 0   127 58 0 0
 [3,] 8940 5474 5848 9587 0 0 0 0 0 0 0 0 0 0    54  0 0 0
 [4,] 8953 5485 5861 9600 0 0 0 0 0 0 0 0 0 0     4  0 0 0
 [5,]   33    0    0    0 0 0 0 0 0 0 0 0 0 0 29870  0 0 0
 [6,] 8928 5467 5844 9577 1 0 0 0 0 1 0 0 0 0    85  0 0 0
```

```
 [7,] 7787 4783 5147 8208 0 0 0 0 0 0 0 0 0 0  3939 39 0 0
 [8,]   48    2   10    7 0 0 0 0 0 0 0 0 0 0 29836  0 0 0
 [9,] 8930 5450 5838 9590 0 0 0 0 0 0 0 0 0 0    27 68 0 0
[10,] 8933 5451 5832 9586 0 0 0 0 0 0 0 0 0 0    45 56 0 0
[11,] 8880 5436 5829 9584 0 0 0 0 0 0 0 0 0 0   121 53 0 0
[12,] 8931 5454 5834 9588 0 0 0 0 0 0 0 0 0 0    40 56 0 0
[13,] 8891 5437 5827 9585 0 0 0 0 0 0 0 0 0 0   102 61 0 0
[14,] 7041 4329 4629 7577 0 0 0 0 0 0 0 0 0 0  6299 28 0 0
[15,] 8952 5484 5858 9602 0 0 0 0 0 0 0 0 0 0     7  0 0 0
[16,] 8906 5442 5814 9567 0 0 0 0 0 0 0 0 0 0   144 30 0 0
[17,] 8931 5450 5839 9591 0 0 0 0 0 0 0 0 0 0    39 53 0 0
[18,] 8926 5439 5838 9602 0 0 0 0 0 0 0 0 0 0    45 53 0 0
[19,] 8913 5440 5829 9572 1 3 6 1 3 1 0 0 0 0    82 52 0 0
[20,] 8919 5458 5835 9564 0 0 0 0 0 0 0 0 0 0   114 13 0 0
```

```
seqs_nt_prop <- seqs_nt_count/rowSums(seqs_nt_count)
print(seqs_nt_prop)
```

```
 [5,] 0.001103568 0.000000e+00 0.0000000000 0.0000000000 0.000000e+00
 [6,] 0.298565361 1.828245e-01 0.1954318965 0.3202688693 3.344146e-05
 [7,] 0.260408655 1.599505e-01 0.1721231983 0.2744875096 0.000000e+00
 [8,] 0.001605190 6.688292e-05 0.0003344146 0.0002340902 0.000000e+00
 [9,] 0.298632244 1.822560e-01 0.1952312477 0.3207036083 0.000000e+00
[10,] 0.298732569 1.822894e-01 0.1950305989 0.3205698425 0.000000e+00
[11,] 0.296960171 1.817878e-01 0.1949302746 0.3205029596 0.000000e+00
[12,] 0.298665686 1.823897e-01 0.1950974819 0.3206367254 0.000000e+00
[13,] 0.297328027 1.818212e-01 0.1948633916 0.3205364010 0.000000e+00
[14,] 0.235461325 1.447681e-01 0.1548005217 0.2533859479 0.000000e+00
[15,] 0.299367956 1.833930e-01 0.1959000769 0.3211049059 0.000000e+00
[16,] 0.297829649 1.819884e-01 0.1944286526 0.3199344547 0.000000e+00
[17,] 0.298665686 1.822560e-01 0.1952646892 0.3207370498 0.000000e+00
[18,] 0.298498478 1.818881e-01 0.1952312477 0.3211049059 0.000000e+00
[19,] 0.298063739 1.819215e-01 0.1949302746 0.3201016620 3.344146e-05
[20,] 0.298264388 1.825235e-01 0.1951309233 0.3198341304 0.000000e+00
                R            W            S            Y            K V H D B
 [1,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
 [2,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
 [3,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
 [4,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
 [5,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
 [6,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 3.344146e-05 0 0 0 0
 [7,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
 [8,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
 [9,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
[10,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
[11,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
[12,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
[13,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
[14,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
[15,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
[16,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
[17,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
[18,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
[19,] 0.0001003244 0.0002006488 3.344146e-05 0.0001003244 3.344146e-05 0 0 0 0
[20,] 0.0000000000 0.0000000000 0.000000e+00 0.0000000000 0.000000e+00 0 0 0 0
                N            - + .
 [1,] 0.0009029194 0.0017723974 0 0
 [2,] 0.0042470655 0.0019396047 0 0
 [3,] 0.0018058389 0.0000000000 0 0
 [4,] 0.0001337658 0.0000000000 0 0
 [5,] 0.9988964318 0.0000000000 0 0
 [6,] 0.0028425242 0.0000000000 0 0
 [7,] 0.1317259138 0.0013042170 0 0
```

```
[13,] 0.0034110290 0.0020399291 0 0
[14,] 0.2106477611 0.0009363609 0 0
[15,] 0.0002340902 0.0000000000 0 0
[16,] 0.0048155703 0.0010032438 0 0
[17,] 0.0013042170 0.0017723974 0 0
[18,] 0.0015048657 0.0017723974 0 0
[19,] 0.0027421998 0.0017389560 0 0
[20,] 0.0038123265 0.0004347390 0 0
```

```
seqs_nt_prop_actg <- apply(seqs_nt_prop, 1, function(x) sum(x[names(x) %in% c("A", "C", "T"
print(seqs_nt_prop_actg)
```

```
 [1] 0.997324683 0.993813330 0.998194161 0.999866234 0.001103568 0.997090593
 [7] 0.866969869 0.002240578 0.996823061 0.996622412 0.994181186 0.996789620
[13] 0.994549042 0.788415878 0.999765910 0.994181186 0.996923386 0.996722737
[19] 0.995017222 0.995752934
```

Result: Filtered Sequences

- 17 sequences left after filtering

```
# keeping sequences with at least 85% of A, C, T, G
seqs_filtered <- seqs[seqs_nt_prop_actg >= 0.85]
print(seqs_filtered)
```

```
DNAStringSet object of length 17:
      width seq                                               names
 [1] 29903 NNNNAAGGTTTATACCTTCCCAG...AAAAAAAAAAAAAAAAAAAAA WHP10644-AV41_RC8...
 [2] 29903 NNNNNNNGTTTATACCTTCCCAG...NNNNNNNNNNNNNNNNNNNNN WHP7426-H2-iseq_1...
 [3] 29903 NNNNNNNNNNNNNNNNNNNNN...AAAAAAAAAAAAAAAAAAAAA WHP3916_10681
 [4] 29903 NNNNAAGGTTTATACCTTCCCAG...AAAAAAAAAAAAAAAAAAAAA WHP1939_3582
 [5] 29903 NNNNNNNNNNNNNNNNNNNNN...AAAAAAAAAAAAAAAAAAAAA WHP3977_10906
 ...   ... ...
[13] 29903 NNNNNNNGTTTATACCTTCCCAG...AAAAAAAAAAAAAAAAAAAAA VOC1980-H2-iseq_NA
[14] 29903 NNNNAAGGTTTATACCTTCCCAG...AAAAAAAAAAAAAAAAAAAAA WHP7692-A4v1_1024583
[15] 29903 NNNNNAGGTTTATACCTTCCCAG...AAAAAAAAAAAAAAAAAAAAA WHP9591-A4v1_1283879
[16] 29903 AYTAAAGGTTTATACCTTCCCAG...AAAAAAAAAAAAAAAAAAAAA WHP6749-H2-iseq_R...
[17] 29903 NNNNNNNNNNNNNNNNNNNNN...AAAAAAAAAAAAAAAAAAAAA WHP4969_12214
```

5. SEQUENCE ANALYSIS

1. Calculate GC content for each of two randomly selected sequences.
2. Extract the spike gene region (positions 21,563 to 25,384) for both sequences.
3. Calculate the codon usage for one of the extracted sequences.

Calculate GC content for each of two randomly selected sequences

```
set.seed(2024) # set seed for reproducibility, in this way you will get the same random num
seqs_sample <- sample(seqs_filtered, 2)
seqs_sample_nt_count <- alphabetFrequency(seqs_sample)
print(seqs_sample_nt_count)
```

```
        A    C    G    T M R W S Y K V H D B   N  - + .
[1,] 8883 5434 5827 9574 0 0 0 0 0 0 0 0 0 0 127 58 0 0
[2,] 8928 5467 5844 9577 1 0 0 0 0 1 0 0 0 0  85  0 0 0
```

```
seqs_sample_nt_count_df <- as.data.frame(seqs_sample_nt_count)
seqs_sample_nt_count_df$length <- rowSums(seqs_sample_nt_count)
seqs_sample_nt_count_df$GC <- seqs_sample_nt_count_df$G + seqs_sample_nt_count_df$C
seqs_sample_nt_count_df$GC_content <- seqs_sample_nt_count_df$GC/seqs_sample_nt_count_df$le
```

```
print(names(seqs_sample))
```

```
[1] "WHP7426-H2-iseq_1019418" "WHP3977_10906"
```

```
print(seqs_sample_nt_count_df$GC_content)
```

```
[1] 0.3765843 0.3782564
```

```
print(paste0("GC content of sequence ", names(seqs_sample)[1], ": ", seqs_sample_nt_count_d
```

```
[1] "GC content of sequence WHP7426-H2-iseq_1019418: 0.376584289201752"
```

```
print(paste0("GC content of sequence ", names(seqs_sample)[2], ": ", seqs_sample_nt_count_d
```

```
[1] "GC content of sequence WHP3977_10906: 0.378256362237903"
```

Extract the spike gene region (positions 21,563 to 25,384) for both sequences

- Visit https://codon2nucleotide.theo.io for positions and annotations of the SARS-CoV-2 genome

```
spike_gene <- subseq(seqs_sample, start=21563, end=25384)
print(spike_gene)
```

```
DNAStringSet object of length 2:
    width seq                                               names
[1]  3822 ATGTTTGTTTTTCTTGTTTTATT...GAGTCAAATTACATTACACATAA WHP7426-H2-iseq_1...
[2]  3822 ATGTTTGTTTTTCTTGTTTTATT...GAGTCAAATTACATTACACMTAA WHP3977_10906
```

Calculate the codon usage for one of the extracted sequences

```
set.seed(20241107)
spike_gene_selected <- spike_gene[sample(1:2, 1)]
length(spike_gene_selected)
width(spike_gene_selected)
```

```
1
3822
```

Using a built-in function to calculate the codon usage

```
codon_usage_auto <- trinucleotideFrequency(spike_gene_selected, step = 3)
print(codon_usage_auto)
```

```
      AAA AAC AAG AAT ACA ACC ACG ACT AGA AGC AGG AGT ATA ATC ATG ATT CAA CAC
[1,]  38  34  23  54  39  10   3  44  20   5  10  17  18  14  14  44  46   4
     CAG CAT CCA CCC CCG CCT CGA CGC CGG CGT CTA CTC CTG CTT GAA GAC GAG GAT
```

```
[1,]  16  13  25   4   0  29   0   1   2   9   9  12   3  36  34  18  14  43
     GCA GCC GCG GCT GGA GGC GGG GGT GTA GTC GTG GTT TAA TAC TAG TAT TCA TCC
[1,]  27   9   2  41  17  15   3  48  15  21  13  48   1  14   0  40  26  12
     TCG TCT TGA TGC TGG TGT TTA TTC TTG TTT
[1,]   2  37   0  12  12  28  28  18  20  59
```

Check the reverse complement of the spike gene

```
spike_gene_selected_revcomp <- reverseComplement(spike_gene_selected)
print(spike_gene_selected_revcomp)
```

```
DNAStringSet object of length 1:
    width seq                                                names
[1]  3822 TTAKGTGTAATGTAATTTGACTC...AATAAAACAAGAAAAACAAACAT WHP3977_10906
```

Translate the spike gene

```
spike_gene_selected_aa <- translate(spike_gene_selected, if.fuzzy.codon="solve")
print(spike_gene_selected_aa)
```

```
AAStringSet object of length 1:
    width seq                                                names
[1]  1274 MFVFLVLLPLVSSQCVNLTTRTQ...SCCKFDEDDSEPVLKGVKLHYT* WHP3977_10906
```

Write the spike gene to a fasta file

```
writeXStringSet(spike_gene_selected, "/content/spike_gene_nt.fasta")
writeXStringSet(spike_gene_selected_aa, "/content/spike_gene_aa.fasta")
```

```
1
```