A sequence alignment and analysis of SARS-CoV-2 spike glycoprotein

Jean-Yves Sgro, Ph.D.

Last Updated: 2020-07-09

Contents

Preamble

In this age of *Next Gen. Sequencing* and complete genomes it is easy to forget about the "earlier days of sequencing" of much smaller portions but complete sequences of genes or proteins.

There are free tools on web servers that can be found to be useful. However, this tutorial is to explore **command-line options** that have the advantage to be scriptable and scalable to a much larger scale and number of files to handle.

Creating this tutorial document became more complex and intricated than anticipated at first.

The original aim of creating just a few examples of sequence alignment became a more elaborate project that:

- provides examples of the power of using simple Unix tools to create powerful manipulations *without* programming
- details methods for sequence retrieval online
- shows examples of 3D structure download without visiting a web site
- helps create automated sequence alignment with command-line tools

In some way this document represents "how I work" and hopefully will be useful or perhaps provide inspiration to even "casual" users.

Coronavirus glycoprotein

As a preamble reading, users could get acquainted with the Spike protein from the "Virology Blog" from Pr. Vincent Racaniello titled: Furin cleavage site in the SARS-CoV-2 coronavirus glycoprotein[1]

> **Summary:** " ...The membrane of coronaviruses harbors a trimeric transmembrane spike (S) glycoprotein which is essential for entry of virus particles into the cell.
>
> The S protein contains two functional domains: a receptor binding domain, and a second domain which contains sequences that mediate fusion of the viral and cell membranes.
>
> The S glycoprotein must be cleaved by cell proteases to enable exposure of the fusion sequences and hence is needed for cell entry.
>
> The spike glycoprotein of the newly emerged SARS-CoV-2 contains a potential cleavage site for furin proteases.
>
> Proteolytic cleavage of the S glycoprotein can determine whether the virus can cross species, e.g. from bats to humans.
>
> Acquisition of the furin cleavage site might be viewed as a 'gain of function' that enabled a bat CoV to jump into humans and begin its current epidemic spread. "

[1]https://www.virology.ws/2020/02/13/furin-cleavage-site-in-the-sars-cov-2-coronavirus-glycoprotein/

Chapter 1

Introduction

This tutorial and exercises are not meant for "*naive beginners*" of the command line interface with *shell* commands rooted in the Linux/Unix platform.

Additionally, to avoid "installation" of various software we'll use the "container" method from `docker` that allows to run Linux software on both Mac and Windows once the `docker` software has been installed. See more details below in section 1.3.1.

Windows users can alternatively install the "*Windows Subsystem for Linux*"[1] to add the relevant command-line functionality which is different than the command-line within Windows itself and have access to a full text-based Linux shell.

NOTE

Beginners should take the time to learn and *Intermediate* users should review the necessary skills and software from these tutorials:

- Survival command-line for Biologists[a]
- Docker – Beginner Biologist[b]

[a]https://bcrf.biochem.wisc.edu/nix-tutorials-survival-command-line/

[b]https://bcrf.biochem.wisc.edu/docker-beginner-for-biologists/

[1]https://docs.microsoft.com/en-us/windows/wsl/install-win10

1.1 Learning goals

My primary goal is to empower users to use the powerful Unix framework to use existing tools for analysis without the need to create specialized software.

I use these tools all the time, even for simple tasks. This is the first time that I am sharing what I do in this fashion within a tutorial. I do rely a lot on search engine research and repositories of questions that others have already asked and experts have answered. This allows me to piece together the string of commands that I am testing for creating an output. There is often more than one answer possible.

1.2 Exercise files

All necessary exercise files can be retrieved rather than created and links will be provided for individual files within relevant chapters.

Alternatively the following "zip" file (compatible Mac/PC) containing all files can be retrieved: `covid19files.zip`.

1.3 Software used during this tutorial

Commands will be issued within a "Text Terminal":

- **Macintosh**: `Terminal` located in `/Applications/Utilities`

- **Windows**: Terminal from "Windows Subsystem for Linux" (see above.)
- **Linux**: Terminal icon might be called "shell."

The following software can be installed or can alternatively be used within `docker` (see below.)

- `EMBOSS` - http://emboss.sourceforge.net/
- `Clustal Omega` - http://www.clustal.org/omega/
- `TCoffee` - http://www.tcoffee.org/

`docker` users have the alternate option to download images rather than install software:

- `docker` - https://docs.docker.com/get-docker/ (free registration required.)
- `docker` *images* used for:
  - `EMBOSS`: https://hub.docker.com/r/pegi3s/emboss
  - `Clustal Omega`: https://hub.docker.com/r/pegi3s/clustalomega
  - `TCoffee`: https://hub.docker.com/r/cbcrg/tcoffee

pegi3s: Bioinformatics Docker Images Project: https://pegi3s.github.io/dockerfiles/

cbcrg: Comparative Bioinformatics *Centro de Regulacio Genomica* (Center for Genomic Regulation) - (Barcelona): https://hub.docker.com/u/cbcrg

Readers can pre-download the docker images with the following commands, assuming that `docker` has been installed.

```
# login
docker login # may not be necessary

# "Pull" the images with the docker pull commands that follow:

docker pull pegi3s/emboss:latest

docker pull pegi3s/clustalomega:latest

docker pull cbcrg/tcoffee
```

Note: These commands can be found on the Docker Hub for each image under the "Tags" tab.

1.3.1 A note about Docker

The `docker` software is a wonderful tool to be able to run Linux software that does not need to be installed on the local, host computer. `docker` *images* are used to create a *container* that will run the software independently of the operating system of the host computer. Even though the software is Linux it can run on Macintosh and Windows as well within the container.

However there are limitations to being able to use that software:

- The CPU has to be new enough to possess "Hyper-V" *virtualization* architechture. Therefore older computers cannot take advantage of this technology.

- There are many *images* available on the "docker hub" but many of them do not offer clear instructions and may not be useable by everyone.

In this tutorial I have used *images* that are functional for the purpose at hand, and I have supplied the **complete commands** to run both the *container* but also the software within. Therefore the reader only need to install the `docker` software to use this option. Alternatively, all commands used within a container would transpose exactly if the used software were to be installed on the local computer.

1.4 Style

In PDF form, command code and software output will appear with a ligh colored background.

Command code will be highlighted within the text: `example` in all document formats.

In the HTML version of this document commands within a `bash` terminal will appear with a light green background and can easily be copied as usual by mouse highlighting, or by using the "copy to clipboard" icon on the right hand side. Example:

```
# This is an example of  bash command
echo HTML version have an icon on the right hand side to copy commands
```

```
HTML version have an icon on the right hand side to copy commands
```

Output in HTML will typically be printed with a gray background as shown above.

Command invoking the `docker` software will be shown with a light blue background. Example:

```
# Command invoking docker software
docker run -it --rm ... etc.
```

Chapter 2

Getting started

To get started you'll need to have access to a computer that allows you to open a **text terminal** and type commands. The relevant information of what is needed can be found in the Introduction, Chapter 1.

Process:

- First we'll search for relevant sequences on NCBI and save them in FASTA format. This will be done *via* a web browser.

- The rest of the analysis will be done using various methods and software from a command-line perspective (see section1.3.)

2.1 Get Sequences

We'll work with the protein sequence to start with.

These preparation steps will be performed within a web browser:

TASK

Open a web browser and follow instructions below:

1. Go to: https://www.ncbi.nlm.nih.gov/protein/ This will take you to the NCBI "protein" database.
2. For a precise research of the Spike protein for only CoV-2 enter the follwing search code within the text field:

```
surface glycoprotein[All Fields] AND "Severe acute respiratory
syndrome coronavirus 2"[Organism]
```

On the date of the first writing on April 16, 2020 the result was a list of `795` proteins.

During revisions on May 26, 2020 the list was `4239` and grew to `5465` on June 18, 2020 and `7847` on June 25, 2020. This change will also impact the numbers for the alternate option to select only unique sequences (see below) from `85` to `942` (May 26) and `1198` (both June 18 and June 25) items.

Many example will keep the shorter list as an example. But commands would transpose to larger sequence lists as well.

A box at the top of the page provides an alternate option:

> **See the results of this search (85 items) in our new Identical Protein Groups database.**

The new Identical Protein Groups database[1]. *The Identical Protein Groups (IPG) resource makes it easier to find protein information by searching against groups of protein records where each group represents a unique protein sequence.*

We'll choose this option to avoid carrying many proteins that are 100% identical.

3. Click on the link: results of this search (85 items.) (That number will increase with time, see remark above.)

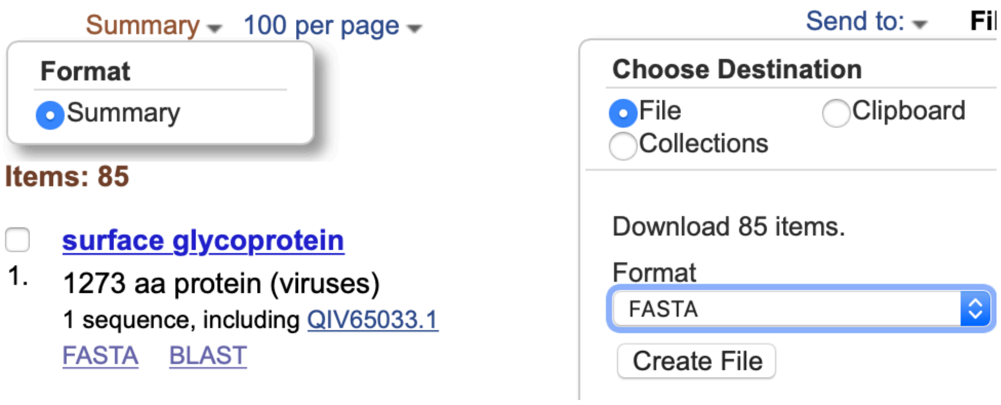
[1]https://www.ncbi.nlm.nih.gov/ipg/docs/about

At the top of the page the it should say **Summary 20 per page** by default. On the same line locate the menu **Send to** which we'll use the save the files:

4. Click on **Send to** and select the **File** option

5. Under Format select **FASTA**

6. Click button **"Create File"**. It will automatically download to your default location, most likely "**Downloads**" within your **user** area. The default file name is **`sequence.fasta.txt`**.



Format: [✓ Summary]

Items: 85

1. surface glycoprotein
1273 aa protein (viruses)
1 sequence, including QIV65033.1

Choose Destination: [✓ File] Clipboard Collections

Download 85 items.

Format: FASTA

Figure 2.1: Details: how to download FASTA files from NBCI page.

Note: The original file with only 85 sequences is available for download as `spike_raw_85.fa`

Note: You can repeat the saving if you wish to download a comma-delimited (`.csv`) file with **all** proteins and their associated nucleic acid codes by selecting "Identical Protein Groups" as the format.

2.2 Getting organized

From this point we'll use line commands. Some steps might be feasible with the mouse, but I encourage you to use the command-line, even for a simple thing as creating a new directory[2].

TASK
Open a text Terminal and follow instructions below:

1. **Start Terminal**: (see section 1.3.)

2. Terminal will look within your User directory by default (*e.g.* `/Users/jsgro` for me.) This is your default "home" directory.
3. If you want to save your project elsewhere, first change to that location, *e.g.* `cd Desktop`:

```
cd ~/Desktop
```

3. Create a new directory:

```
mkdir spike
```

4. Change into this directory:

```
cd spike
```

5. Move the saved sequence file saved with the browser here. By default the file would be saved in the Downloads folder of your user area. Of course if your browser saved the file elsewhere then adapt to that location. You can take this opportunity to rename the file while moving it from its download location.

Since there were 85 files we could choose *e.g.* `spike_85.fa` but most likely this number will evolve with time. So perhaps `spike_raw.fa` would more generic and endure better with time. (See update in section 2.1.)

```
mv ~/Downloads/sequence.fasta.txt spike_raw.fa
```

[2]"Directory and Folder" are equivallent.

2.3 Checking sequences

There are 2 things we want to see:

1.Are there any “**partial**” sequences. The complete sequence is 1273 in length. 2. Are there any **X** wihtin the sequence, meaning that the amino acid sequence is not 100% complete as an **X** represents an *unknown* or *undefined* amino acid, due to uncertainty within the nucleotide sequencing.

We can quickly accomplish these tasks with the command grep that recognizes a *pattern*. In our case the patterns will be either the word **partial** or a capital **X**.

Checking for “partial” with fgrep (*fast* grep as it only works with simple patterns.) We just need to provide the pattern followed by the file name. The -i option makes the command case insensitive. We only print the first three sequence names.

```
fgrep -i partial spike_raw.fa | head -3
```

```
>QJR94977.1 surface glycoprotein, partial [Severe acute respiratory syndrome coronavirus 2]
>QJR93825.1 surface glycoprotein, partial [Severe acute respiratory syndrome coronavirus 2]
>QJR92925.1 surface glycoprotein, partial [Severe acute respiratory syndrome coronavirus 2]
```

Note: You can also use more or less command to inspect the the complete spike_raw.fa file one screen and scroll one screenfull at a time with space bar, and q to quit. We’ll remove these sequences later for the final set after we check for X and retain only the last 3 lines on the screen with tail:

```
fgrep -i X spike_raw.fa  | tail -3
```

```
LLALHRSYLTPGDSXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXRFPNITNLCPFGEVFNATRFASVYAWNRKRISNCVADYSVLYNSASFSTFKCYGVSPTKLNDLCFTNVYADSF
NGVEGFNCYFPLQSYGFQPTNGVGYQPYRVVVLSFELLHAPATVCGPXKSTNLVKNKCVNFNFNGLTGTGVLTESNKKFL
```

Since the name of the sequence does not get retained we do not know which sequence contain the Xs, but there are some for sure.

In the next section we’ll use “piped” command-lines to remove sequences with X and sequences that are “partial”.

Chapter 3

Complete sequences dataset

We now have a set of sequences that are "unique" sine they were obtained from the *Identical Protein Groups* but some of them are "partial" and others contain one or more `X` of undetermined amino acids.

In the next section we'll prepare a "dataset" that only contains sequences that are "complete" (full length) and do not contain any `X`.

The ***method*** will be to use *existing Unix utilities* to achieve this task, without a mouse or a graphical interface. Most importantly, there is no manual editing to remove the sequence. Therefore if or when there is an update at NCBI increasing the number of downloadable "unique" sequences, the steps below can be repeated as a "script" and without the possibility to introduce errors as might be the case if it were done manually.

The reader is encouraged to review or learn about the commands and concepts below, see suggested masterials in Introduction, Chapter 1.

The following Unix utilities will be used:

- **`cat`** - print file(s) onto the screen (standard output)
- **`sed`** - stream editor - modify data "on the fly" *e.g.* substiute one or mores

character string for another.
- `tr` - translate characters (*i.e.* replace or delete)
- `fgrep`: fast `grep` - file pattern searcher

The following Unix key concepts will be used:

- **Standards**: standard input, standard output
- **Data "*streams*"**: "*redirection*" and "*piping*" of "standard input/output"

The following symbols will be used:

- `|` - "pipe"symbol, to *receive* or *pass* the "standard" stream of data from the *previous* or to the *next* command.
- `>` - "redirect" final standard output into the named file.

3.1 Removing partial and sequences with `X`

In order to remove the "unwanted" sequences, those that are "partial" or contain one or more `X`, we'll take advange of the format of the file containing all the "unique" sequences `spike_raw.fa`. The file is in a multi-sequence "FastA" format[1].

We will want to remove files that:

- contain the word "partial" within the description line.

- contain any number of `X` within the sequence data.

In FastA format each sequence starts with a "greater than" symbol > followed immediately by the sequence name. Everything after the first blank space is considered "annotation". Subsequent lines collectively make up the (protein) sequence. A new line with the > symbol marks the beginning of a new, separate sequence.

Below we can see the first 2 lines of the first 2 sequences within the file. The `-A1` qualifier shows one line *after* the line containing the searched pattern >. (The `--` marks separate output results.)

[1]https://en.wikipedia.org/wiki/FASTA_format

```
grep -A1 ">" spike_raw.fa  | head -6
```

```
>QJU70245.1 surface glycoprotein [Severe acute respiratory syndrome coronavirus 2]
MFVFLVLXPLVSSQCVNLTTRTQLPPAYTNSFTRGVYYPDKVFRSSVLHSTQDLFLPFFSNVTWFHAIHVSGTNGTKRFD
--
>QJU70329.1 surface glycoprotein [Severe acute respiratory syndrome coronavirus 2]
MFVFLVLLPLVSSQCVNLTTRTQLPPAYTNSFTRGVYYPDKVFRSSVLHSTQDLFLPFFSNVTWFHAIHVSGTNGTKRFD
--
```

"***Algorithm***":

Here is a proposed method to find and edit out "unwanted" sequences from the multi-sequence file. It is not necessary the most efficient or the most elegant, but has the advantage of using existing tools and does not require any programming.

We'll use "word processing" methods to achieve the goal with "Unix utilities" used for replacing and substituting characters, including "new line" (return character.)

Each protein sequence will be converted from multiple lines containging name, annotation and sequence into a *single line* containing all information and sent within the "data stream" pipeline.

At that point the `grep` command can simply "weed-out" the lines (hence the sequences) that contain the word "partial" or the letter `X`.

Finally, the "return characters" are added again to recreate to the original, multi-line format.

Here is a "final" command that will accomplish the task. Results are saved into a file: **`spike_filtered.fa`**.

The command below is "split" after the pipe character `|` to allow writing one command per line for more clarity:

The reader is encouraged to add each command after the `|` pipe sign *one at a time* press return and observe the effect it has...

Hint: optionally add `head` to avoid long outputs.

```
cat spike_raw.fa |
  sed -e 's/>/*>/g' -e 's/$/#/g' |
  tr -d '\n' |
  tr '*' '\n' |
  fgrep -v partial |
  fgrep -v X  |
  tr '#' '\n' |
  sed 1d > spike_filtered.fa
```

Here is an explanation of each line:

- `cat spike_raw.fa`: send the content of file into the data stream (standard input/output.)
- `sed -e 's/>/*>/g' -e 's/$/#/g'`: substitute `>` into `*>` and execute (`-e`) another coommand to exchange "end of line" (represented by `$`) with `#`. This will be used later to re-establish the multiline format.
- `tr -d '\n'` - translate utility: delete all "new-lines" (return characters) in essence transforming the whole into a single line.
- `tr '*' '\n'` - re-establish a return **before** the > character: now each sequence and its name and annotation is represented into a single line.
- `fgrep -v partial` and `fgrep -v X`: fast grep recongizes the patterns and inverts the selection (`-v`) to retain *only* the sequences that don't match these patterns. They are the sequences we want to keep.
- `tr '#' '\n'`: convert each # character into a "new line" (return character.) This re-establishes the multi-line format for the sequence.
- `sed 1d` removes the first line which would be just a return character introduced in the previous step when replacing # character with "new line."
- `> spike_filtered.fa`: redirect standard output (with symbol >) into a file named `spike_filtered.fa`.

Command Design Note:

The `cat` command on the first line could be omitted if the file name was provided after the first `sed` command as the input argument. However, writing the command this way may make it easier for some readers to better understand how the “data stream” is initiated. In addition the “flow” of commands is only from left to right if started with `cat`.

3.2 Counting sequences

We can count how many sequences “survived” the “purge”: it will be equal to the number of lines that contain the symbol > wihtin the file, that we can count with the Unix utility “word count” `wc` and requesting only the number of lines with `-l` added:

```
fgrep ">" spike_filtered.fa | wc -l
```

```
32
```

Upon first revision this number grew to `167`. Larger numbers are expected as more sequences are added to the database (see 2.1.)

3.3 Evaluating sequence length

When we downloaded the sequences from NCBI it seemed that many of them had a length of `1273`.

All complete sequences are probably of almost the same length but as an exercise we can try to evalulate the length of the first sequence within the filtered file using Unix utilities.

Utilities used:

- `head`: shows the first 10 lines or designated amount of lines.
- `sed`: stream editor. Used to remove top line(s).

- `tr`: used to delete return characters as they would be counted by `wc`.
- `wc`: word count. Provides number of lines, words, characters (including returns.) Option `-m` only show number of characters.

```
head -17 spike_filtered.fa  | sed 1d | tr -d '\n' | wc -m
```

```
1273
```

Command Design Note:

The first line provides the name of the sequence and needs to be removed therefore keeping only sequence records of amino acids. Return characters are removed as they would be counted.

Note: On preparing file `spike_filtered.fa` the last line was `sed 1d` to remove a return character. Without this step the above command would have to be modified to accommodate the extra line and written as:

```
head -18 spike_filtered.fa  | sed 1,2d | tr -d '\n' | wc
-m
```

In the next section we'll align sequences.

Note: The original filtered sequence file of 32 sequences is available for download as `spike_32.fa` which could be used instead of the `spike_filtered.fa` just generated.

Chapter 4

Sequence alignment

A simple web browser search could easily locate many *multiple sequence alignment* (**MSA**) web servers. But our purpose here is to find ways to easily "do the analysis again" in a "reproducible way" and "without manual input" for all the various analysis steps, that could even ideally be placed within a script.

The spike protein sequences within file `spike_filtered.fa` are distinct just by one to a handful amino acids, and therefore most MSA program would have no difficulty aligning them.

We'll use one of the latest MSA software: *clustal omega*[1] (Sievers et al. (2011)) in a command-line version.

Clustal omega can be downloaded and installed but it is also available as a docker container, therefore avoiding all the installation process. (See Introduction Chapter 1 for material suggestion to learn how to use `docker`.)

Whether you use a `docker` container or a locally installed verion the commands should remain the same. Here we'll start a container to access the local directory.

[1]http://www.clustal.org/omega/

4.1 Run clustal omega

For help type `clustalo --help`.

Assuming that you have installed a local version of the software and are looking within the directory containing the filtered sequence file, run `clustalo` with the following command with input `-i`, output `-o`, and verbose `-v` options:

```
clustalo -i spike_filtered.fa -o spike_filtered_omega.fa -v
```

To run from within a docker container the following command can be used: (shown with command continuation \ for clarity.)

```
docker run -it --rm  \
-v $(pwd):/data -w /data \
pegi3s/clustalomega \
-i spike_filtered.fa -o spike_filtered_omega.fa -v
```

Command Design Note:

This is a typical `docker` command that will `run` in an interactive terminal (`-it`) within a container that will be removed upon completion of the task (`--rm`.)

The current directory `$(pwd)` is mapped (`-v`) to a directory named `/data` that will be created within the container and set as the default working directory (`-w`.)

The pulled docker image used to create the temporary container is named `pegi3s/clustalomega` and its internal installation of *clustalomega* (`clustalo` - implied) will immediately run upon the starting of the container and is provided with the input `-i`, output `-o` commands and files that should be present in the working directory. The verbose (`-v`) command will provide explicit information as the `clustalo` program runs

Docker for WINDOWS:

The variable defining the current directory `$(pwd)` is created on the fly in a Unix/Linux/MacOS environment.

Windows users would need one more step and use curly brackets:

```
# step 1 - define variable with Get-Location command:
$loc = Get-Location

# step 2: implement docker command with curly brackets
# e.g. within PowerShell or cmd Windows terminal:

docker run -it --rm  -v ${loc}:/data -w /data pegi3s/clustalomega
-i spike_filtered.fa -o spike_filtered_omega.fa -v
```

Therefore the `docker run` command only differs by replacing `$(pwd)` with the predifined variable `${loc}` written within curly brackets rather than parenthesis.

Note that a Windows PATH could be used instead of the variable, for example `C:\Users\someone\somewhere\`.

In either case the following output will be echoed on the terminal thanks to the verbose option. The number of threads will depend on your CPU.

```
Using 4 threads
Read 32 sequences (type: Protein) from spike_filtered.fa
not more sequences (32) than cluster-size (100), turn off mBed
Calculating pairwise ktuple-distances...
Ktuple-distance calculation progress done.
CPU time: 0.56u 0.02s 00:00:00.58 Elapsed: 00:00:00
Guide-tree computation done.
Progressive alignment progress done.
CPU time: 5.95u 0.68s 00:00:06.63 Elapsed: 00:00:07
```

```
Alignment written to spike_filtered_omega.fa
```

Since the sequences are very similar, looking through the aligned sequences file does not seem to provide much insight at first glance. For example using the command:

```
more spike_filtered_omega.fa
```

Changing the format from Multiple FastA format where sequences are shown one by one sequentially to a format where sequences are "meshed", "interleaved", or "interlaced" together in an actual alignment might be helpful.

For this we can use the `EMBOSS`[2] software that have been developped over the years to provide tools pertinent to (old fashioned) sequence analysis.

As with Clustal Omega, `EMBOSS` can be installed locally or accessed as a `docker` container. The latter is the easiest option. (See Introduction Chapter 1 for material suggestion to learn how to use `docker`.)

4.2 Alignment format

Amongst the many sequence and multiple sequence formats available[3] for `EMBOSS` one of the simplest interleaved format is the "clustal" option.

The `EMBOSS` program used to manipulate the format of sequence files is called `seqret`. (The list of `EMBOSS` "apps" is available online[4].)

The format can be specified simply by adding the format name before the sequence file itsef (as a "prefix",) both separated by a double colon `::`.

For example, assuming that you have `EMBOSS` installed locally, the format change from multiple FastA format to the `clustal` format would be written as:

[2]http://emboss.sourceforge.net/
[3]http://emboss.sourceforge.net/docs/themes/SequenceFormats.html
[4]http://emboss.sourceforge.net/apps/release/6.6/emboss/apps/

```
seqret \
fasta::spike_filtered_omega.fa clustal::spike_filtered_omega.clustal
```

To perform this task with a docker container (shown with command continuation symbol \ for clarity.)

```
docker run -it --rm                      \
-v $(pwd):/data  -w /data                \
pegi3s/emboss                            \
seqret fasta::spike_filtered_omega.fa    \
clustal::spike_filtered_omega.clustal
```

Upon completion the user is returned to the local shell and the container is discarded. (Windows users can refer to section 4.1 above for specific docker for Windows command format.)

Docker Magic

The first 3 lines of the `docker run` command above create a new container. The `seqret` command and subsequent lines can simply be changed to alternate `EMBOSS` commands that are shown below.

Other interleaved sequence format options could be used in the same way, for example `msf`, `nexus`, `phylip` etc.

However, again, upon inspection of the interleaved sequence file, it is still difficult to spot if there is any difference or where differences are located between the files.

The format `mega` is useful in this case as only the top sequence will be shown in full, while only differences will be displayed for the remaining sequences.

```
seqret fasta::spike_filtered_omega.fa mega::spike_filtered_omega.mega
```

Here is a command to skip some of the top header to look at the first 5 lines os sequences:

```
head -15 spike_filtered_omega.mega | tail -5
```

```
#QIU81885.1     MFVFLVLLPLVSSQCVNLTTRTQLPPAYTNSFTRGVYYPDKVFRSSVLHS
#QIU80913.1     .................................................L
#QIU81585.1     ..................................................
#QIU80973.1     ..........................V.......................
#QIS61422.1     ..................................................
```

This is indeed a useful format visually. In the next section we'll discover that we can also add a consensus sequence and count the number of amino acid changes.

4.3 Consensus sequence

A consensus sequence can be useful in some cases. The `EMBOSS` program `showalign` can take a multiple FastA sequence and present it in a format similar ot the `mega` format just showing differences with an added consensus sequence. Other options can show *dissimilarities*, *similarities*, *identities*, *non-identities*, etc.

To show *dissimilarities*:

```
showalign -show=d spike_filtered_omega.fa spike_filtered_omega.showalign
```

We can check the result with:

```
head -7 spike_filtered_omega.showalign
```

```
                         10        20        30        40        50        60
                ----:----|----:----|----:----|----:----|----:----|----:----|
QIU81885.1      ............................................................
QIU80913.1      .................................................L..........
QIU81585.1      ............................................................
QIU80973.1      ..........................V.................................
QIS61422.1      ............................................................
```

Note: Your results may be different as the number of sequences augment with time and no changes might be visible in the portion shown.

A consensus sequence is visible as the last sequence, here shown at the very end:

```
tail -4 spike_filtered_omega.showalign
```

```
QIJ96493.1      .............
QII57278.1      .............
Consensus       SEPVLKGVKLHYT
```

The computation of the consensus sequence can be modulated with optional parameter `plurality` and its 50% default value and other options affecting the similarity calculations based on the chosen "scoring matrix." Other options can be used to influence the aspect with uppercase/lowercase, ruler etc.

Other `EMBOSS` apps can also be used to calculate consensus or display the alignment in graphical formats.

Explore

Try the following `EMBOSS` apps suggested below, varying the options shown here as example that avoid user manual input:

- `prettyplot  -boxcol -consensus -cidentity grey -graph png spike_filtered_omega.fa`
- `/usr/lib/emboss/cons spike_filtered_omega.fa  -out consensus.fasta`

Note: in some installation `cons` is not readily available unless full path is given.

Chapter 5

Distance matrix

The sequences are very similar to each other as we could observe in the alignment.

But how many amino acids are different between the various sequences?

Another questions we could askis "what is the largest number of differences amongst all the sequences?"

The calculation of a "*distance matrix*" could help, and `clustalo` can calculate such a matrix while performing the alignment.

The qualifier `--force` is only necessary if the calculation needs to be run multiple times (*e.g.* when testing) to allow the overwriting of a previous file.

```
clustalo -i spike_filtered.fa -o spike_filtered_omega.fa -v  \
--distmat-out=spike_filtered_omega.dist \
--full  --force
```

Or if using `docker`: (Windows users can refer to section 4.1 for specific Windows command format.)

```
docker run -it --rm -v $(pwd):/data -w /data \
pegi3s/clustalomega -i spike_filtered.fa -o spike_filtered_omega.fa -v \
--distmat-out=spike_filtered_omega.dist \
--full --force
```

We can look at the text file of the matrix with the following command that will prevent "soft wrapping" of lines:

```
less -S spike_filtered_omega.dist
```

Here we print a few truncated lines to explore the format showing the first 4 lines and the first 60 characters of each line:

```
cut -c 1-60  < spike_filtered_omega.dist | head -4
```

```
32
QIU81885.1      0.000000 0.001571 0.001571 0.001571 0.001571
QIU80913.1      0.001571 0.000000 0.001571 0.001571 0.001571
QIU81585.1      0.001571 0.001571 0.000000 0.001571 0.001571
```

In this case 32 is the number of sequences and is shown alone on the first line. (Current update now has 167 sequences.)

But these numbers are not very useful in themselves.

5.1 Convert to number of differences matrix

We could also have calculated the values as a percentage by adding `--percent-id`. However, instead of `0.001571` we would have for example `99.842891`, a value close to 100 as there are very few differences indeed.

In both cases we can make use of the matrix by multiplying these numbers by the sequence length since all sequences are of length 1273.

With the appropriate "rouding" to the next integer we would get the following for default and percent versions of the matrix:

- `0.001571 * 1273 = 1.999883` which is rounds up to **2**
- `(99.842891/100) * 1273 = 1271`, and `1271 -  1273 = 2` as well

In other words, *if we multiply each number within the matrix we'll obtain the number of differing amino acids between each pairwise sequence comparison.*

The following is an advanced command that I modified based on the answer to a similar question on an Internet forum[1]. The purpose of the command is to apply the multiplicated example above to all numbers within the matrix.

```
awk '{ for ( i=2; i<=NF; i++ ) printf int($i*1273) " " } { print $1," " }' \
spike_filtered_omega.dist > spike_diff.dist
```

Briefly it is a `for` loop within an `awk` script:

- `i=2`: start with second column. First column contains sequence names
- `i<=NF`: as long as `i` is less than `NF` (number of fields or columns)
- `i++`: then increment `i` by a value of `1` at each round.
- `printf`: is a formatted print output
- `int`: is the `awk` command to "round" numbers
- `$i`: represents the row of numbers. All will be multiplied by `1273`
- `" "`: is part of the `printf` formatting to add a blank space between each number. There is one blank space between the quotes.
- `{ print $1,"" }`:
  - `" "` represents each modified line of the previous section *i.e* the line of calculated numbers.
  - `$1` adds column 1 with sequence names of original matrix. However, it ends up at the end of the line.

The output is a set of small one digit numbers representing the number of amino acid differences between each sequence pair.

A diagonal of `0` values indicate the comparison of files with themselves.

Below is a complete matrix output when the results contained only 32 sequences.

At a glance we could immediately conclude that the sequence that is most different to all others is `QHS34546.1` located on the one before last line with `5` or even

[1]https://www.linuxquestions.org/questions/linux-general-1/awk-multiply-fields-with-constant-4175440426/ ; archived April 25, 2020 https://bit.ly/3aCdcDJ

6 differences with all other sequences.

```
# Full matrix when there were only 32 sequences

cat spike_diff.dist
```

```
32
0 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIU81885.1
1 0 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIU80913.1
1 1 0 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIU81585.1
1 1 1 0 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIU80973.1
1 1 1 1 0 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIS61422.1
1 1 1 1 1 0 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIS61338.1
1 1 1 1 1 1 0 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIS61254.1
1 1 1 1 1 1 1 0 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIS60930.1
1 1 1 1 1 1 1 1 0 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIS60978.1
1 1 1 1 1 1 1 1 1 0 1 1 1 3 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 5 1 QIS60906.1
1 1 1 1 1 1 1 1 1 1 0 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIS60546.1
1 1 1 1 1 1 1 1 1 1 1 0 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIS60489.1
1 1 1 1 1 1 1 1 1 1 1 1 0 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIS60582.1
3 3 3 3 3 3 3 3 3 3 3 3 3 0 3 1 1 3 1 3 3 3 3 3 3 3 3 3 3 3 6 3 QIS30615.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 0 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 5 1 QIS30425.1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIK50427.1
3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 1 0 3 1 3 3 3 3 3 3 3 3 3 3 3 6 3 QIS30295.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 0 1 1 3 1 1 1 1 1 1 1 1 1 5 1 QIS30335.1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 4 1 YP_009724390.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 0 3 1 1 1 1 1 1 1 1 1 5 1 QIS30165.1
3 3 3 3 3 3 3 3 3 1 3 3 3 3 1 3 3 3 1 3 0 3 3 3 3 3 3 3 3 3 6 3 QIQ49882.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 0 1 1 1 1 1 1 1 1 5 1 QIO04367.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 0 1 1 1 1 1 1 1 5 1 QIC53204.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 0 1 1 1 1 1 1 5 1 QII87830.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 0 1 1 1 1 1 5 1 QHU79173.2
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 0 1 1 1 1 5 1 QIA98583.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 0 1 1 1 5 1 QIA20044.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 0 1 1 5 1 QIJ96493.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 0 1 5 1 QII57278.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 0 5 1 QHR84449.1
5 5 5 5 5 5 5 5 5 5 5 5 5 6 5 5 6 5 4 5 6 5 5 5 5 5 5 5 5 5 0 5 QHS34546.1
1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 3 1 1 1 3 1 1 1 1 1 1 1 1 1 5 0 QHZ00379.1
```

At last revision the number of sequences retained has increased to `167`.

A quick visual inspection of the updated file is easily accomplished with the command `less -S` to avoid soft wrapping:

```
less -S spike_diff.dist
```

This quick glance shows that sequence `QHS34546.1` *appears* to still be the most different. It *seems* to also contain the higest value of difference (currently `9`, stars added) roughly in the middle of the matrix results for this sequence:

```
6 8 6 6 4 7 7 6 6 6 5 6 6 6 6 6 6 6 6 6 6 5 5 5 6 6 5 6 6 6 6 5
6 6 6 5 6 6 6 6 7 5 6 5 6 6 6 5 6 5 5 6 5 4 5 6 6 6 6 7 6 6 5 6
6 5 5 5 5 5 6 5 7 5 5 6 6 6 6 7 7 6 5 6 6 5 6 6 6 6 6 5 6 7 6
*9* 5 5 5 5 7 6 5 8 6 5 6 1 6 5 5 6 5 5 6 5 6 5 5 5 6 6 5 5 6 7
6 6 6 5 5 6 6 6 6 5 5 5 6 6 5 5 5 5 6 5 7 5 6 5 5 5 5 6 6 5 5 6
5 0 5 5 5 5 5 5 5 QHS34546.1
```

The sentence above is written with the words *appears* and *seems* in order to specify that this statement is based on (human) casual visual inspection.

But can we "automatically" find the highest number of differences without visual inspection and *without writing a complicated program*?

5.2 Find largest value in matrix

The result file `spike_diff.dist` can be considered as a matrix of numbers if we except the first line with the single number of sequences and the last column containing the sequence names. In computer programming the fancy name would be "*array*."

A simple search with the terms **`find largest number in array`** on a popular search engine provides the answer: *About 202,000,000 results* and therefore many pages with solutions in various programming languages are available: `Python`, `C`, `C++`, `javascript`, `java`, `swift`, `ruby` and many more.

However, to answer the question "what is the largest number in the array" (or matrix) we can, once more, use a pipeline of simple Unix tools.

Algorithm:

Here is a solution that does not require any programming and calls on very simple, almost "ordinary" Unix tools:

- remove the first line that contains the number of sequences
- remove the last column (or its content) that lists the sequence names
- convert the blanks space between the matrix numbers to return characters
- sort the resulting single column of numbers numerically, keeping only unique values, and reverse the output order so that the largest is at the top of the results.

And now let's do it! The following two commands achieve the same goal:

```
# version 1
sed 1d spike_diff.dist | tr ' ' '\n'| fgrep -v . | sort -u -b -r

# version 2
sed 1d spike_diff.dist  | tr ' ' '\n'| grep -v [A-Z] | sort -u -n -r
```

- `sed 1d spike_diff.dist`: All versions start by deleting the first line (`1d`) of file `spike_diff.dist` with the stream editor `sed` and send the remaining data within the data stream (pipeline.)

- `tr ' ' '\n'`: convert all blanks into a return character in all versions: this will convert the matrix into a single column of numbers (that can easily be sorted.)

- In *version 1* with command `fgrep -v \.` we take advantage to the fact that the matrix only contains integer numbers and that sequence names always end with a period followed by a number. Therefore the pattern "." finds only sequence names and the qualifier `-v` inverses the pattern and retains only those lines that *do not* contain the pattern. The command `fgrep` is a special version of `grep` for which the command should be written as `grep -v \.`

instead. The "`\.`" notation "escapes" the dot with help of a back-slash \ as an "actual dot" rather than "any character" since `grep` uses "regular expressions" to encode patterns. The caveat with this method is that it would not remove any sequence name that does not contain a dot.

- In *version 2* with command `grep -v  [A-Z]` we use the "regular expression" pattern recognition of `grep` and remove all matching lines with qualifier `-v`. Here the pattern [A-Z] represents any uppercase letter. To make the case more general and include lower case letters we could add one piped command for lower case letters `grep -v  [a-z]` or combine both with yet another version of grep: `egrep -v  "[A-Z]|[a-z].`
- `sort -u -n -r` is a sorting utility for which we add flad to sort unique items `-u`, sort based on numerical rather than alphabetical properties with `-b` and reverse the order with `r` so that the largest number be listed at the top.
- `sort -n`: The end of version 3

The result of any of these commands is currently:

```
9
8
7
6
5
4
3
1
0
```

To recuperate only the top value we could add `head -1`.

Even simpler: we don't in fact need to remove the sequence names if we sort numerically. *Version 3* uses the `tail` command to print only the last (and largest number) located on the last line.

```
# version 3
sed 1d spike_diff.dist | tr ' ' '\n' | sort -n | tail -n1
```

5.3 Alternatives

There are many alternatives that can be found with a search engine. However, they are usually more complicated than the "simple algorithm" we used above. Incidently I also found a few online answers that did not give the correct answer. Hence testing results is essential.

Here are 2 mini "programs" found online[2] that provided the correct answer with a file (x.) containing only the matrix of numbers (*i.e.* removed first line and last column.)

```
# Prepare x.:  remove 1rst line and last column
sed 1d spike_diff.dist | awk '{$NF=""; print}'  > x.
```

Awk version:

```
awk '$1 > m || NR == 1 { m = $1 } END { print m }' x.
```

Perl version:

```
perl -MList::Util=max -alne '$tmp = max @F; $max = $tmp if $max < $tmp; END { print $max }' x.
```

Note: The commands to prepare `x.` and the `perl` or `awk` commands could all be stringed together with the pipe symbol to create a single pipeline. For example for the *awk version* we would string commands together as:

```
sed 1d spike_diff.dist | awk '{$NF=""; print}' | awk '$1 > m || NR == 1 { m = $1 } END { print m }'
```

Finding the simplest and least specific solution is usually the best answer as it can be used in multiple settings. The simplest solution "*version 3*" above is probably the best answer to finding the largest number.

[2]https://unix.stackexchange.com/questions/130899/finding-the-maximum-of-the-values-in-a-file

Chapter 6

Align related sequences

Recent papers provide in-depth analysis of the spike protein on the base of structure, sequence, and computation (Walls et al. (2020), Wang et al. (2020), Lokman et al. (2020)) amongst the vast number of released papers.

Aligning multiple sequences that are longer can be challenging even for sequences that are relatively similar. A few decades ago manual adjustments were inevitable. Nowadays software with combined powerful algorithms and methods can give very good results. Human intervention by manual adjustments might still be needed in some areas of lesser similarity, but it may be that we just cannot know.

This section is meant to explore how to align a sequence of the SARS_CoV-2 spike glycoprotein S to other related spike sequences based on examples in the cited papers.

To this aim one could use web services but that is not the goal of this tutorial. For those interested in web services, the aligners proposed on this page are worth checking: https://bip.weizmann.ac.il/toolbox/structure/seq_align.htm (Archived: 13OCT2016 https://bit.ly/37SlLKV)

For the exercise below we'll use TCoffee (Notredame et al. (2000), Thompson (2009)).

6.1 Spike protein features

Before attempting an alignment of the SARS_CoV-2 spike protein to other similar structure let's look at a summary of the the spike protein sequence features. Figure 1 of Lokman et al. (2020) provides a graphical comparison between SARS_CoV and SARS-CoV-2 spike sequence features.

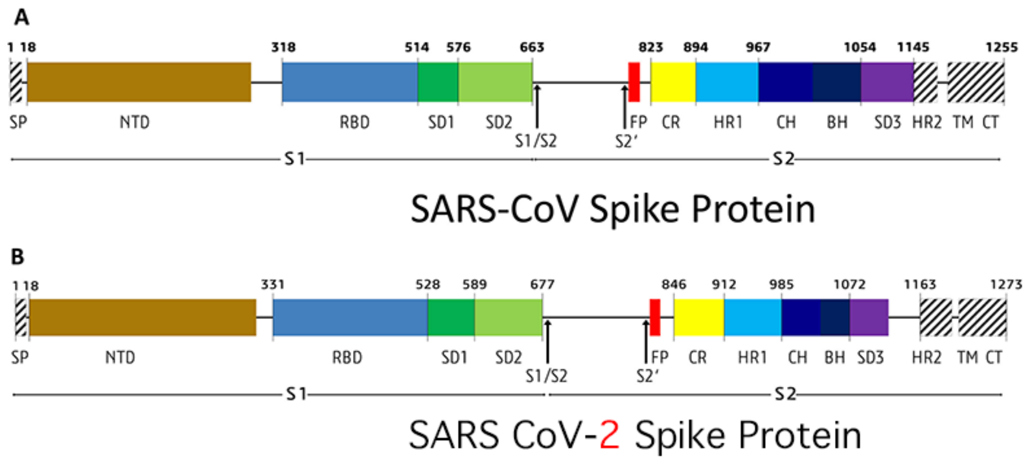


Figure 6.1: SARS-CoV and CoV-2 Spike protein.

Figure 6.1 legend: SP= signal peptide, NTD= N-terminal domain, RBD= receptor-binding domain, SD1= subdomain 1, SD2= subdomain 2, S1/S2= S1/S2 protease cleavage site, S2'= S2' protease cleavage site, FP= fusion peptide, HR1= heptad repeat 1, CH= central helix, CH= central helix, BH= β-hairpin, HR2= heptad repeat 2, TM= transmembrane domain, CT= cytoplasmic tail. Arrows denote protease cleavage sites.

SARS-CoV-2 has emerged with remarkable properties that include a novel, unique furin cleavage site (P**RRAR**↓SV) at S1/S2 boundary in the S spike glycoprotein.

The role of the sequence features of the spike protein is elegantly summarized by Lokman et al. (2020):

> “Viral entry to the host cell is initiated by the receptor-binding domain (RBD) of S1 head. Upon receptor-binding, proteolytic cleavage occurs at S1/S2 cleavage site and two heptad repeats (HR) of S2 stalk form a six-helix bundle structure triggering the release of the fusion peptide. As it comes into close proximity to the transmembrane anchor (TM), the TM domain facilitates membrane destabilization required for fusion between virus-host membranes.”

6.2 Related sequences

We will attempt to reproduce the full-lentgh alignment presented in Walls et al. (2020) supplementary material “Data S1.”[1] We will limit the exercise to creating the best possible automatic alignment without using manual editing.

The sequences aligned in the paper are listed in the following table showing accession number and short name used within the supplemental full length alignment.

Table 6.1: Spike Glycoprotein S accession codes

Accession code	Short name
YP_009724390.1	**SARS-CoV-2**
QHR63300.2	SARSr-CoV_RaTG13
AAP13441.1	SARS-CoV_Urbani
AAP13567.1	SARS-CoV_CUHK-W1
AAS00003.1	SARS-CoV_GZ02
AAV97988.1	SARS-CoV_A031
AAV91631.1	SARS-CoV_A022
ALK02457.1	WIV16
AGZ48828.1	WIV1
AVP78042.1	SARSr-CoV_ZXC21
AVP78031.1	SARSr-CoV_ZC45

[1]The link to the full-length sequence aligment is not trivial to find. It can be found at https://bit.ly/3dthDBS and was archived at https://bit.ly/2zXNP2R.

Accession code	Short name
Q3I5J5.1	SARSr-CoV_Rp3
ACU31032.1	SARSr-CoV_Rs672

NOTE: a link for direct download will be provided in the exercise section 6.3 below.

We'll use TCoffee (Thompson (2009)) to create the alignment. TCoffee is a complex, multi-algorithm system that can also take advantage on online database searching. The online version can be accessed at http://www.tcoffee.org/ and contains multiple options to align sequences. Here are a few options listed on the web site:

- T-Coffee Aligns DNA, RNA or Proteins using the default T-Coffee
- M-Coffee Aligns DNA, RNA or Proteins by combining the output of popular aligners
- Expresso Aligns protein sequences using structural information
- PSI-Coffee Aligns distantly related proteins using homology extension (slow and accurate)

PSI-Coffee (Definition) uses the EBI web-services and runs remotely (at the EBI) the BLASTs required for the homology extension procedure.

Expresso (Definition) is the most accurate mode of T-Coffee and creates structure-based alignments. Expresso fetches PDB structures whose similarity to the original sequence is higher than 30% (by default) that can be used as a template

In the exercise below we'll use a line-command in effect is combining Expresso and PSI-Coffee.

Since TCoffee is complex and complicated to install the exercise below will be presented in `docker`. Readers that do not readily have access to `docker` should test the possibilities on the TCoffee web site.

The T-Coffee Server is hosted by the Centre for Genomic Regulation (CRG) of Barcelona.

6.3 Create the alignment

In this section we'll download relevant files and use the TCoffee software to create the alignment. We'll add structure files to the PSI-Coffee method in order to provide information for structural alignment and annotations.

- **Sequence files**: we'll use the same sequences that appear in Walls et al. (2020)
- **Structure files**: Protein Data Bank files 3D coordinates

6.3.1 Step 1: download sequence files

TASK
Download the sequence files listed in table @ref(tab:"Spike Glycoprotein S accession codes") from section 6.2 above.

Files should be saved in the simple "fasta" format within a single text file.

Option 1: Retrieve previously saved sequences in a ready-to-use file: sarbecos.fasta

Option 2: Use NCBI "Batch Entrez" with the *Protein* database.

Create a list with accession files, for example using the following (Copy/Paste) code taken from the table above to create a text file named `sarbecos.list`.

```
echo "YP_009724390.1
QHR63300.2
AAP13441.1
AAP13567.1
AAS00003.1
AAV97988.1
AAV91631.1
ALK02457.1
AGZ48828.1
```

```
AVP78042.1
AVP78031.1
Q3I5J5.1
ACU31032.1" > sarbecos.list
```

Alternatively download a premade list file: sarbecos.list

Then use that list on the Batch Entrez web site (see right hand side red arrow on figure 6.2.)

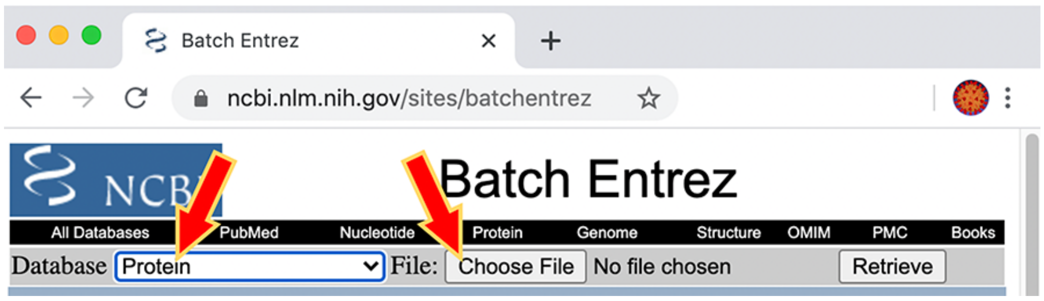


Figure 6.2: Details: Batch Entrez. Select Protein database and upload file list of accession codes.

Batch Entrez Instructions from the "Batch Entrez" web site:

"Given a file of Entrez accession numbers or other identifiers, Batch Entrez downloads the corresponding records."

1. Start with a local file containing a list of accession numbers or identifiers

2. Select the database corresponding to the type of accession numbers or identifiers in your input file

3. Use the **Browse** or **Choose File...** button to select the input file

4. Press the **Retrieve** button to see a list of document summaries

5. Select a format in which to display the data for viewing, and/or saving (*Choose fasta*)
6. Select 'Send to file' to save the file.

6.3.2 Step 2: download structure files

We'll download two Protein Data Bank (PDB) files in PDB format for the complete spike protein with code 6VXX and 6VYB. Since some portions of these structures are missing (too flexible to be dectected) we'll add 2 other structures for the "*receptor-binding domain complexed with its receptor ACE2*" for SARS-CoV-2: 6LZG, and for SARS-CoV: 2AJF.

Note: for simplicity PDB files could be omitted.

Files can be downloaded from the web site or from the following command lines with either `wget` ("Web get" - not installed on MacOS by default) or `curl` ("Copy URL" - requires redirect):

```
# Commands with curl - (Copy URL)
curl http://files.rcsb.org/download/6VXX.pdb > 6vxx.pdb
curl http://files.rcsb.org/download/6VYB.pdb > 6vyb.pdb
curl https://files.rcsb.org/download/6LZG.pdb > 6lzg.pdb
curl https://files.rcsb.org/download/2AJF.pdb > 2ajf.pdb

# Alternate wget commands - (Web get)
wget http://files.rcsb.org/download/6VXX.pdb
wget http://files.rcsb.org/download/6VYB.pdb
wget https://files.rcsb.org/download/6LZG.pdb
wget https://files.rcsb.org/download/2AJF.pdb
```

Please not the exact writing (upper/lower case) of the files as they are saved.

Files should be saved in the same directory as the sequence file save previously.

6.3.3 Step 3: create alignment

Here we'll use `docker` with a Docker image issued by the TCoffee authors at https://hub.docker.com/r/cbcrg/tcoffee from the Comparative Bioinformatics *Centro de Regulacio Genomica* (Centre for Genomic Regulation) hence *CBCRG* group. There is no information on the Docker Hub about the image itself but it is fully functional and its automated maintenance is detailed on the Release Building Procedure page.

TCoffee needs to be connected to the Internet to access the BLAST server on EBI and PSI databases. Therefore the `docker` command needs to provide a bridge to the Internet connection of the local computer. This is accomplished with the `--net=host` option. Other options mean: `-it`= interactive and terminal; `--rm`= delete the container when the job is done; `-v`= provide access to the current directory and map it to `/data` within the container; `-w`= define the default working directory within the container.

TASK
Make sure that the terminal is set to the directory containing both the sequence and structure files. Use `pwd` and `ls` to verify that this is the case.

We'll now "plunge into" the docker container.
The prompt will change from `$` to `#`.

Note: On HTML version of this document (but not PDF) we'll be reminded that we are *within the container* by the blueish background behind the lines of command code.

```
docker run -it --rm --net=host -v $(pwd):/data -w /data cbcrg/tcoffee
```

The files used are named:

- `sarbecos.fasta`: multiple sequence file in fasta format in the desired order for final output.
- `*.pdb`: 3D coordinate files in PDB format.

TCoffee command to compute the alignment used:

```
t_coffee sarbecos.fasta -outorder=input -seqnos \
-pdb 6vxxA 6vybA 2ajfE 6lzgB  -mode psicoffee
```

Command Details:

- `t_coffee`: activate the TCoffee software
- `sarbecos.fasta`: multiple sequence file
- `-outorder=input`: keep the order of sequences as listed within the fasta file
- `-seqnos`: provide numbering on final output
- `-pdb 6vxxA 6vybA 2ajfE 6lzgB`: last letter added to PDB codes represents chain ID
- `-mode psicoffee`: choose running mode

6.3.4 Results

The complete alignment is shown in Appendix A.

Compute time:

On a Macbook Pro with 4 cores the computation will take about 20 minutes. Most of this time is dedicated to waiting for the live database connections as the reported local CPU time was only 1.22 second.

When using TCoffee *via* a web server[2] it *may* take 48hrs or more to obtain a result.

[2]http://www.tcoffee.org/Projects/tcoffee/

Chapter 7

Alignment Results

The results of the alignment of complete spike glycoprotein sequences is shown in Appendix A in Clustal format as plain text.

The authors of the full-lentgh alignment presented in Walls et al. (2020) supplementary material "Data S1" do not provide specific explanations about the computation of the alignment and if there was any manual, human intervention.

Even though it is not mentioned in the manuscript it is most probable that their supplemental figure[1] was created from the original text alignment through the ESPript server: "*'Easy Sequencing in PostScript', is a program which renders sequence similarities and secondary structure information from aligned sequences for analysis and publication purpose.*"

7.1 Color alignment

***OPTIONAL* TASK**
Create a colored version similar to the Walls paper following instructions

[1]The link to the full-length sequence aligment is not trivial to find. It can be found at https://bit.ly/3dthDBS and was archived at https://bit.ly/2zXNP2R.

below. This is done in 2 steps:

1- Optional: remove the PDB sequences to better match the Walls paper. See command below.
2- Go to the ESPript server: http://espript.ibcp.fr to create output.

The results obtained from our TCoffee alignment can also be converted to this nice rendering and we can additionally include protein structure information in the final output.

7.1.1 Remove PDB sequences (optional)

In order to better match the Walls paper, we'll remove the PDB files that were added before converting to this format. To accomplish this we can use the command below.

```
fgrep -v 6VXX sarbecos.aln | \
fgrep -v 6VYB | fgrep -v 2AJF | \
fgrep -v 6LZG > sarbecos-4.aln
```

Command Design info:

The `.aln` format is simple and we can simply remove the lines containing the code of the PDB sequences. The `fgrep` will recognize these lines and the flag `-v` will ask to remove them. The result of the first command is "piped" into the next until all removals are satisfied. The final output is redirected into a file.

Note: Alternate methods could be used, for example using egrep to remove two patterns at the same time: `egrep -v "6VXX|6VYB" sarbecos.aln | egrep -v "2AJF|6LZG"`. Other solutions can use `awk` (*e.g.* `awk '!/6VXX/' sarbecos.aln` etc.)

7.1.2 Create colored output

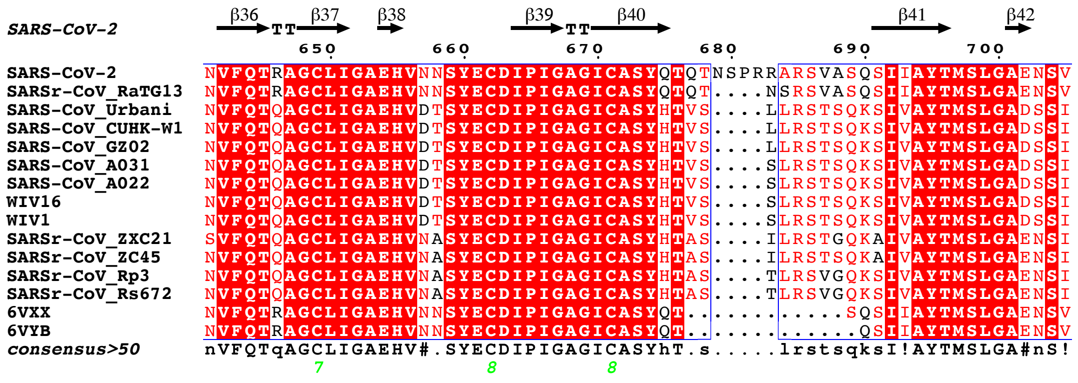
Within the ESPript server:

- use alignment file `sarbecos-4.aln`
- provide a PDB structure code for structure annotation (6VYB)
- specify `chain A`

7.2 Furin site

Th spike glycoprotein contains many features (section 6.1). We'll just take a look at the results for the novel furin recognition site at residue 682. The furin recognition sequence is **RRAR**

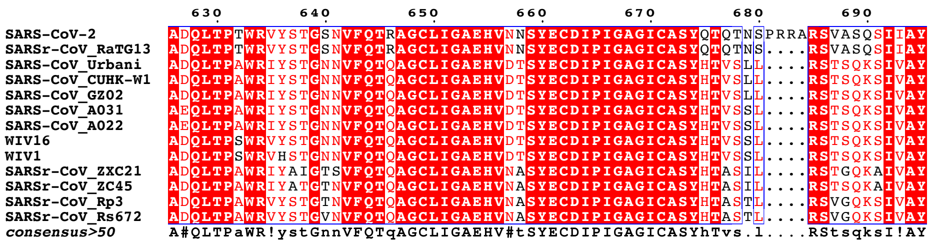
The alignment at this location appears different between our automated TCoffee version and the Walls version.



```
                     β36   β37  β38          β39   β40                   β41     β42
SARS-CoV-2          ---->TT---> ->          ---->TT---->                ----->   ->
                          650       660       670       680       690       700
SARS-CoV-2          NVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQTQTNSPRRARSVASQSIIAYTMSLGAENSV
SARSr-CoV_RaTG13    NVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQTQT....NSRSVASQSIIAYTMSLGAENSV
SARS-CoV_Urbani     NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS....LLRSTSQKSIVAYTMSLGADSSI
SARS-CoV_CUHK-W1    NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS....LLRSTSQKSIVAYTMSLGADSSI
SARS-CoV_GZ02       NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS....LLRSTSQKSIVAYTMSLGADSSI
SARS-CoV_A031       NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS....SLRSTSQKSIVAYTMSLGADSSI
SARS-CoV_A022       NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS....SLRSTSQKSIVAYTMSLGADSSI
WIV16               NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS....SLRSTSQKSIVAYTMSLGADSSI
WIV1                NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS....SLRSTSQKSIVAYTMSLGADSSI
SARSr-CoV_ZXC21     SVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTAS....ILRSTGQKAIVAYTMSLGAENSI
SARSr-CoV_ZC45      NVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTAS....ILRSTSQKAIVAYTMSLGAENSI
SARSr-CoV_Rp3       NVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTAS....TLRSVGQKSIVAYTMSLGAENSI
SARSr-CoV_Rs672     NVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTAS....TLRSVGQKSIVAYTMSLGAENSI
6VXX                NVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQT..........SQSIIAYTMSLGAENSV
6VYB                NVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQT...........QSIIAYTMSLGAENSV
consensus>50        nVFQTqAGCLIGAEHV#.SYECDIPIGAGICASYhT.s.....lrstsqksI!AYTMSLGA#nS!
                          7                8          8
```

Figure 7.1: Alignment details around the furin site for the automated TCoffee alignment. Note the structure information at the top line and the consensus sequence at the bottom line.

Figure 7.1 depicts the result of our automated alignment and figure 7.2 that of the Walls paper. It is indeed unfortunate, but not unexpected, that this region is not visible (hence absent) from the PDB sequences since the sequences are likely cleaved allowing too much flexibility to the cut ends.



```
                     630       640       650       660       670       680       690
SARS-CoV-2        ADQLTPTWRVYSTGSNVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQTQTNSPRRARSVASQSIIAY
SARSr-CoV_RaTG13  ADQLTPTWRVYSTGSNVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQTQTNS....RSVASQSIIAY
SARS-CoV_Urbani   ADQLTPAWRIYSTGNNVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSLL....RSTSQKSIVAY
SARS-CoV_CUHK-W1  ADQLTPAWRIYSTGNNVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSLL....RSTSQKSIVAY
SARS-CoV_GZ02     ADQLTPAWRIYSTGNNVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSLL....RSTSQKSIVAY
SARS-CoV_A031     AEQLTPAWRIYSTGNNVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSSL....RSTSQKSIVAY
SARS-CoV_A022     AEQLTPAWRIYSTGNNVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSSL....RSTSQKSIVAY
WIV16             ADQLTPSWRVYSTGNNVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSSL....RSTSQKSIVAY
WIV1              ADQLTPSWRVHSTGNNVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSSL....RSTSQKSIVAY
SARSr-CoV_ZXC21   ADQLTPAWRIYAIGTSVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTASIL....RSTGQKAIVAY
SARSr-CoV_ZC45    ADQLTPAWRIYATGTNVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTASIL....RSTSQKAIVAY
SARSr-CoV_Rp3     ADQLTPAWRVYSTGTNVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTASTL....RSVGQKSIVAY
SARSr-CoV_Rs672   ADQLTPAWRVYSTGVNVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTASTL....RSVGQKSIVAY
consensus>50      A#QLTPaWR!ystGnnVFQTqAGCLIGAEHV#tSYECDIPIGAGICASYhTvs.l....RStsqksI!AY
```

Figure 7.2: Alignment details around the furin site for Walls (2020) alignment. Note the consensus sequence at the bottom line.

A PyMOL (Schrödinger, LLC (2020)) illustration of this region is shown in figure 7.3. The script used to create the image can be found in appendix B. The inset is simply a zoomed out version of the same. The last visibe residues on each strand are labeled. Residues 677 to 689 have not been resovled.

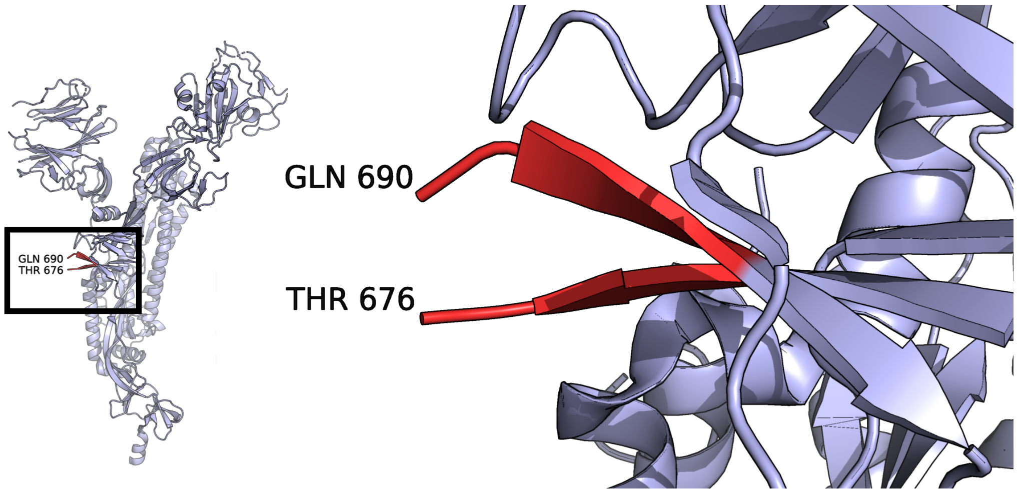


Figure 7.3: PDB ID 6VYB, one chain of the trimeric spike protein showing the missing amino acids around the novel furin cleavage site.

In figure figure 7.2 the four amino acids that *appear* to be extra above a column of dots are PRRA while in figure 7.1 they appear as NSPR.

In addition, a TCoffee Expresso run on the web site a few days ago gave a slight different result in this area as well, the “floating” four amino acids were SPRR.

```
SARS-CoV-2          641 NVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQTQTNSPRRARSVASQSIIA  694
SARSr-CoV_RaTG1     641 NVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQTQTN----SRSVASQSIIA  690
SARS-CoV_Urbani     627 NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSL----LRSTSQKSIVA  676
SARS-CoV_CUHK-W     627 NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSL----LRSTSQKSIVA  676
SARS-CoV_GZ02       627 NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSL----LRSTSQKSIVA  676
SARS-CoV_A031       627 NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSS----LRSTSQKSIVA  676
SARS-CoV_A022       627 NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSS----LRSTSQKSIVA  676
WIV16               627 NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSS----LRSTSQKSIVA  676
WIV1                628 NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVSS----LRSTSQKSIVA  677
SARSr-CoV_ZXC21     617 SVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTASI----LRSTGQKAIVA  666
SARSr-CoV_ZC45      618 NVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTASI----LRSTSQKAIVA  667
SARSr-CoV_Rp3       613 NVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTAST----LRSVGQKSIVA  662
SARSr-CoV_Rs672     613 NVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTAST----LRSVGQKSIVA  662

cons                649 .****:**********: ****************:* :      **...::*:*  702
```

The 3D structure does not help us resolve these conflicts, but it is rather easy to see that moving 2 columns of amino acids from the automated TCoffee alignment just made or one column of the web Expresso version to the left would reproduce the Walls paper version. This could be accomplished with a manual editor that allow easy editing of alignments such as Jalview.

Overall TCoffee Expresso run on the web (not shown) gave a score of “Good” to most of the sequences over their length providing each of these sequences with a score between 97 and 99 with an average score of 98 out of 100.

7.3 Alignment conclusion

In conclusion sequence alignment has made a lot of progress but some incertainty areas may not be resolved without further information. Most of these poorly defined areas are usually loops or coils that are subject to evolutionary pressure

Chapter 8

Acknowledgments

8.1 Licensed icons used:

Exercises / Homework

Icons made by prosymbols available on Flaticon.com.

Direct download: https://www.flaticon.com/free-icon/homework_748646

Study

"Study" icon original from http://www.onlinewebfonts.com/icon is licensed by `CC BY 3.0`.

Icon direct download link: http://cdn.onlinewebfonts.com/svg/download_532202.png

Version used: colorized with PhotoShop by JYS in herits `CC BY 3.0` license.

8.2 This document

Created with R and RStudio using bookdown and a modified rstudio4edu-book template.

Output saved in HTML and PDF formats. While HTML was the primary output focus, care was taken to insure the readability of the PDF version.

In both versions links to web sites are “live”.

Appendix A

Full sequence alignments

Results frrom TCoffee run within docker with command:

```
t_coffee sarbecos.fasta -outorder=input -seqnos \
-pdb 6vxxA 6vybA 2ajfE 6lzgB  -mode psicoffee
```

Note: the alignment is reported here in Clustal format. Other miscellaneous output is not shown.

```
CLUSTAL FORMAT for T-COFFEE Version_13.41.123.92238f3 [http://www.tcoffee.org]
[MODE: psicoffee ], CPU=1.22 sec, SCORE=990, Nseq=17, Len=1283

SARS-CoV-2         MFVFLV-L--LPLVSS----QCVNLTTRTQLPPAYTNSFTRGVYYPDKVF 43
SARSr-CoV_RaTG13   MFVFLV-L--LPLVSS----QCVNLTTRTQLPPAYTNSSTRGVYYPDKVF 43
SARS-CoV_Urbani    MFIFLL-F--LTLTSGSDLDRCTTFDDVQAPNYTQHTSSMRGVYYPDEIF 47
SARS-CoV_CUHK-W1   MFIFLL-F--LTLTSGSDLDRCTTFDDVQAPNYTQHTSSMRGVYYPDEIF 47
SARS-CoV_GZ02      MFIFLL-F--LTLTSGSDLDRCTTFDDVQAPNYTQHTSSMRGVYYPDEIF 47
SARS-CoV_A031      MFIFLL-F--LTLTSGSDLDRCTTFDDVQAPNYTQHTSSMRGVYYPDEIF 47
SARS-CoV_A022      MFIFLL-F--LTLTSGSDLDRCTTFDDVQAPNYTQHTSSMRGVYYPDEIF 47
WIV16              MFIFLF-F--LTLTSGSDLESCTTFDDVQAPNYPQHSSSRRGVYYPDEIF 47
WIV1               MKLLVLVF--ATLVSSYTIEKCLDFDDRTPPANTQFLSSHRGVYYPDDIF 48
SARSr-CoV_ZXC21    MLFFLF-LQFALVNSQCDLTGRTPL----NP--NYTNSSQRGVYYPDTIY 43
```

```
SARSr-CoV_ZC45     MLFFLF-L-----QFALVNSQCVNLTGRTPLNPNYTNSSQRGVYYPDTIY 44
SARSr-CoV_Rp3      MKILIL-A--FLASLAKAQEGCGIISRKPQPKMAQVSSSRRGVYYNDDIF 47
SARSr-CoV_Rs672    MKVLIV-L--LCLGLVTAQDGCGHISTKPQPLMDKFSSSRRGVYYNDDIF 47
6VXX               ---------------------------------AYTNSFTRGVYYPDKVF 17
6VYB               ---------------------------------AYTNSFTRGVYYPDKVF 17
2AJF               -------------------------------------------------- 0
6LZG               -------------------------------------------------- 0


SARS-CoV-2         RSSVLHSTQDLFLPFFSNVTWFHAIHVSGTNGTKRFDNPVLPFNDGVYFA 93
SARSr-CoV_RaTG13   RSSVLHLTQDLFLPFFSNVTWFHAIHVSGTNGIKRFDNPVLPFNDGVYFA 93
SARS-CoV_Urbani    RSDTLYLTQDLFLPFYSNVTGFHTINHT-------FGNPVIPFKDGIYFA 90
SARS-CoV_CUHK-W1   RSDTLYLTQDLFLPFYSNVTGFHTINHT-------FDNPVIPFKDGIYFA 90
SARS-CoV_GZ02      RSDTLYLTQDLFLPFYSNVTGFHTINHT-------FDNPVIPFKDGIYFA 90
SARS-CoV_A031      RSDTLYLTQDLFLPFYSNVTGFHTINHT-------FDNPVIPFKDGIYFA 90
SARS-CoV_A022      RSDTLYLTQDLFLPFYSNVTGFHTINHT-------FDNPVIPFKDGIYFA 90
WIV16              RSDTLYLTQDLFLPFYSNVTGFHTINHR-------FDNPVIPFKDGVYFA 90
WIV1               RSNVLHLVQDHFLPFDSNVTRFITFGLN-------FDNPIIPFKDGIYFA 91
SARSr-CoV_ZXC21    RSDTLVLSQGYFLPFYSNVSWYYSLTTNNAAT-KRTDNPILDFKDGIYFA 92
SARSr-CoV_ZC45     RSDTLVLSQGYFLPFYSNVSWYYSLTTNNAAT-KRTDNPILDFKDGIYFA 93
SARSr-CoV_Rp3      RSNVLHLTQDYFLPFDSNLTQYFSLNVDSDRF-TYFDNPILDFGDGVYFA 96
SARSr-CoV_Rs672    RSDVLHLTQDYFLPFDTNLTRYLSFNMDSATK-VYFDNPTLPFGDGIYFA 96
6VXX               RSSVLHSTQDLFLPFFSNVTWFHAIH----------DNPVLPFNDGVYFA 57
6VYB               RSSVLHSTQDLFLPFFSNVTWFHAI------------HPVLPFNDGVYFA 55
2AJF               -------------------------------------------------- 0
6LZG               -------------------------------------------------- 0


SARS-CoV-2         STEKSNIIRGWIFGTTLDSKTQSLLIVNNATNVVIKVCEFQFCNDPFLGV 143
SARSr-CoV_RaTG13   STEKSNIIRGWIFGTTLDSKTQSLLIVNNATNVVIKVCEFQFCNDPFLGV 143
SARS-CoV_Urbani    ATEKSNVVRGWVFGSTMNNKSQSVIIINNSTNVVIRACNFELCDNPFFAV 140
SARS-CoV_CUHK-W1   ATEKSNVVRGWVFGSTMNNKSQSVIIINNSTNVVIRACNFELCDNPFFAV 140
SARS-CoV_GZ02      ATEKSNVVRGWVFGSTMNNKSQSVIIINNSTNVVIRACNFELCDNPFFAV 140
```

```
SARS-CoV_A031      ATEKSNVVRGWVFGSTMNNKSQSVIIINNSTNVVIRACNFELCDNPFFVV 140
SARS-CoV_A022      ATEKSNVVRGWVFGSTMNNKSQSVIIINNSTNVVIRACNFELCDNPFFVV 140
WIV16              ATEKSNVVRGWVFGSTMNNKSQSVIIINNSTNVVIRACNFELCDNPFFAV 140
WIV1               ATEKSNVIRGWVFGSTMNNKSQSVIIMNNSTNLVIRACNFELCDNPFFVV 141
SARSr-CoV_ZXC21    ATEHSNIVRGWIFGTTLDNTSQSLLIVNNATNVIIKVCNFDFCYDPYLSG 142
SARSr-CoV_ZC45     ATEHSNIIRGWIFGTTLDNTSQSLLIVNNATNVIIKVCNFDFCYDPYLSG 143
SARSr-CoV_Rp3      ATEKSNVIRGWIFGSTFDNTTQSAVIVNNSTHIIIRVCNFNLCKEPMYTV 146
SARSr-CoV_Rs672    ATEKSNVVRGWIFGSTMDNTTQSAIIVNNSTHIIIRVCYFNLCKEPMYAI 146
6VXX               STEKSNIIRGWIFGTTLDSKTQSLLIVNNATNVVIKVCEFQFCNDPFLGV 107
6VYB               STEKSNIIRGWIFGTTLDSK--SLLIVNNATNVVIKVCEFQFCNDPFLGV 103
2AJF               -------------------------------------------------- 0
6LZG               -------------------------------------------------- 0


SARS-CoV-2         ---YYHKNNKSWMESEFRVYSSANNCTFEYVSQPFLMDLEGKQGNFKNLR 190
SARSr-CoV_RaTG13   ---YYHKNNKSWMESEFRVYSSANNCTFEYVSQPFLMDLEGKQGNFKNLR 190
SARS-CoV_Urbani    ---SK----PMGTQTHTMIFDNAFNCTFEYISDAFSLDVSEKSGNFKHLR 183
SARS-CoV_CUHK-W1   ---SK----PMGTQTHTMIFDNAFNCTFEYISDAFSLDVSEKSGNFKHLR 183
SARS-CoV_GZ02      ---SK----PMGTQTHTMIFDNAFNCTFEYISDAFSLDVSEKSGNFKHLR 183
SARS-CoV_A031      ---SK----PMGTQTHTMIFDNAFNCTFEYISDAFSLDVSEKSGNFKHLR 183
SARS-CoV_A022      ---SK----PMGTQTHTMIFDNAFNCTFEYISDAFSLDVSEKSGNFKHLR 183
WIV16              ---SK----PTGTQTHTMIFDNAFNCTFEYISDSFSLDVAEKSGNFKHLR 183
WIV1               ---LK----SNNTQIPSYIFNNAFNCTFEYVSKDFNLDLGEKPGNFKDLR 184
SARSr-CoV_ZXC21    YYHNN----KTWSIREFAVYSFYANCTFEYVSKSFMLNISGNGGLFNTLR 188
SARSr-CoV_ZC45     YYHNN----KTWSIREFAVYSSYANCTFEYVSKSFMLNISGNGGLFNTLR 189
SARSr-CoV_Rp3      ---SR------GAQQSSWVYQSAFNCTYDRVEKSFQLDTAPKTGNFKDLR 187
SARSr-CoV_Rs672    ---SN------EQHYKSWVYQNAYNCTYDRVEQSFQLDTAPQTGNFKDLR 187
6VXX               ------------------------NCTFEYVS-------------FKNLR 120
6VYB               -------------------------CTFEYVS-------------FKNLR 115
2AJF               -------------------------------------------------- 0
6LZG               -------------------------------------------------- 0
```

```
SARS-CoV-2          EFVFKNIDGYFKIYSKHTPINLVRDLPQGFSALEPLVDLPIGINITRFQT 240
SARSr-CoV_RaTG13    EFVFKNIDGYFKIYSKHTPINLVRDLPPGFSALEPLVDLPIGINITRFQT 240
SARS-CoV_Urbani     EFVFKNKDGFLYVYKGYQPIDVVRDLPSGFNTLKPIFKLPLGINITNFRA 233
SARS-CoV_CUHK-W1    EFVFKNKDGFLYVYKGYQPIDVVRDLPSGFNTLKPIFKLPLGINITNFRA 233
SARS-CoV_GZ02       EFVFKNKDGFLYVYKGYQPIDVVRDLPSGFNTLKPIFKLPLGINITNFRA 233
SARS-CoV_A031       EFVFKNKDGFLYVYKGYQPIDVVRDLPSGFNTLKPIFKLPLGIKITNFRA 233
SARS-CoV_A022       EFVFKNKDGFLYVYKGYQPIDVVRDLPSGFNTLKPIFKLPLGIKITNFRA 233
WIV16               EFVFKNKDGFLYVYKGYQPIDVVRDLPSGFNILKPIFKLPLGINITNFRA 233
WIV1                EFVFRNKDGFLHVYSGYQPISAASGLPTGFNALKPIFKLPLGINITNFRT 234
SARSr-CoV_ZXC21     EFVFRNVDGHFKIYSKFTPVNLNRGLPTGLSVLQPLVELPVSINITKFRT 238
SARSr-CoV_ZC45      EFVFRNVDGHFKIYSKFTPVNLNRGLPTGLSVLQPLVELPVSINITKFRT 239
SARSr-CoV_Rp3       EYVFKNRDGFLSVYQTYTAVNLPRGLPIGFSVLRPILKLPFGINITSYRV 237
SARSr-CoV_Rs672     EYVFKNKDGFLSVYNAYSPIDIPRGLPVGFSVLKPILKLPISINITSFKV 237
6VXX                EFVFKNIDGYFKIYSKHTPINLVRDLPQGFSALEPLVDLPIGINITRFQT 170
6VYB                EFVFKNIDGYFKIYSKHTPINLVRDLPQGFSALEPLVDLPIGINITRFQT 165
2AJF                -------------------------------------------------- 0
6LZG                -------------------------------------------------- 0


SARS-CoV-2          LLALHRSYLTPGDSSSGWTAGAAAYYVGYLQPRTFLLKYNENGTITDAVD 290
SARSr-CoV_RaTG13    LLALHRSYLTPGDSSSGWTAGAAAYYVGYLQPRTFLLKYNENGTITDAVD 290
SARS-CoV_Urbani     ILTAFSP------AQDIWGTSAAAYFVGYLKPTTFMLKYDENGTITDAVD 277
SARS-CoV_CUHK-W1    ILTAFSP------AQDTWGTSAAAYFVGYLKPTTFMLKYDENGTITDAVD 277
SARS-CoV_GZ02       ILTAFLP------AQDTWGTSAAAYFVGYLKPTTFMLKYDENGTITDAVD 277
SARS-CoV_A031       ILTAFSP------AQGTWGTSAAAYFVGYLKPTTFMLKYDENGTITDAVD 277
SARS-CoV_A022       ILTAFSP------AQGTWGTSAAAYFVGYLKPTTFMLKYDENGTITDAVD 277
WIV16               ILTAFLP------AQDTWGTSAAAYFVGYLKPATFMLKYDENGTITDAVD 277
WIV1                LLTAFPP------RPDYWGTSAAAYFVGYLKPTTFMLKYDENGTITDAVD 278
SARSr-CoV_ZXC21     LLTIHRGD---PMSNNGWTAFSAAYFVGYLKPRTFMLKYNENGTITDAVD 285
SARSr-CoV_ZC45      LLTIHRGD---PMPNNGWTAFSAAYFVGYLKPRTFMLKYNENGTITDAVD 286
SARSr-CoV_Rp3       VMAMFSQ------TTSNFLPESAAYYVGNLKYTTFMLSFNENGTITNAID 281
SARSr-CoV_Rs672     VMSMFSR------TTSNFLPEVAAYFVGNLKYSTFMLNFNENGTITDAID 281
6VXX                LLALH-----------------AAYYVGYLQPRTFLLKYNENGTITDAVD 203
```

```
6VYB              LL--------------------AAYYVGYLQPRTFLLKYNENGTITDAVD 195
2AJF              -------------------------------------------------- 0
6LZG              -------------------------------------------------- 0


SARS-CoV-2        CALDPLSETKCTLKSFTVEKGIYQTSNFRVQPTESIVRFPNITNLCPFGE 340
SARSr-CoV_RaTG13  CALDPLSETKCTLKSFTVEKGIYQTSNFRVQPTDSIVRFPNITNLCPFGE 340
SARS-CoV_Urbani   CSQNPLAELKCSVKSFEIDKGIYQTSNFRVVPSGDVVRFPNITNLCPFGE 327
SARS-CoV_CUHK-W1  CSQNPLAELKCSVKSFEIDKGIYQTSNFRVVPSGDVVRFPNITNLCPFGE 327
SARS-CoV_GZ02     CSQNPLAELKCSVKSFEIDKGIYQTSNFRVVPSRDVVRFPNITNLCPFGE 327
SARS-CoV_A031     CSQNPLAELKCSVKSFEIDKGIYQTSNFRVVPSGDVVRFPNITNLCPFGE 327
SARS-CoV_A022     CSQNPLAELKCSVKSFEIDKGIYQTSNFRVVPSGDVVRFPNITNLCPFGE 327
WIV16             CSQNPLAELKCSVKSFEIDKGIYQTSNFRVAPSKEVVRFPNITNLCPFGE 327
WIV1              CSQNPLAELKCSVKSFEIDKGIYQTSNFRVAPSKEVVRFPNITNLCPFGE 328
SARSr-CoV_ZXC21   CALDPLSETKCTLKSLSVQKGIYQTSNFRVQPTQSIVRFPNITNVCPFHK 335
SARSr-CoV_ZC45    CALDPLSETKCTLKSLTVQKGIYQTSNFRVQPTQSVVRFPNITNVCPFHK 336
SARSr-CoV_Rp3     CAQNPLAELKCTIKNFNVSKGIYQTSNFRVSPTQEVIRFPNITNRCPFDK 331
SARSr-CoV_Rs672   CAQNPLSELKCTIKNFNVSKGIYQTSNFRVSPTHEVIRFPNITNRCPFDK 331
6VXX              CALDPLSETKCTLKSFTVEKGIYQTSNFRVQPTESIVRFPNITNLCPFGE 253
6VYB              CALDPLSETKCTLKSFTVEKGIYQTSNFRVQPTESIVRFPNITNLCPFGE 245
2AJF              ---------------------------------------------CPFGE 5
6LZG              ------------------------------------------TNLCPFGE 8
                                                               *** :

SARS-CoV-2        VFNATRFASVYAWNRKRISNCVADYSVLYNSASFSTFKCYGVSPTKLNDL 390
SARSr-CoV_RaTG13  VFNATTFASVYAWNRKRISNCVADYSVLYNSTSFSTFKCYGVSPTKLNDL 390
SARS-CoV_Urbani   VFNATKFPSVYAWERKKISNCVADYSVLYNSTFFSTFKCYGVSATKLNDL 377
SARS-CoV_CUHK-W1  VFNATKFPSVYAWERKKISNCVADYSVLYNSTFFSTFKCYGVSATKLNDL 377
SARS-CoV_GZ02     VFNATKFPSVYAWERKRISNCVADYSVLYNSTFFSTFKCYGVSATKLNDL 377
SARS-CoV_A031     VFNATKFPSVYAWERKRISNCVADYSVLYNSTSFSTFKCYGVSATKLNDL 377
SARS-CoV_A022     VFNATKFPSVYAWERKRISNCVADYSVLYNSTSFSTFKCYGVSATKLNDL 377
WIV16             VFNATTFPSVYAWERKRISNCVADYSVLYNSTSFSTFKCYGVSATKLNDL 377
WIV1              VFNATTFPSVYAWERKRISNCVADYSVLYNSTSFSTFKCYGVSATKLNDL 378
```

```
SARSr-CoV_ZXC21   VFNATRFPSVYAWERTKISDCIADYTVFYNSTSFSTFKCYGVSPSKLIDL 385
SARSr-CoV_ZC45    VFNATRFPSVYAWERTKISDCIADYTVFYNSTSFSTFKCYGVSPSKLIDL 386
SARSr-CoV_Rp3     VFNATRFPNVYAWERTKISDCVADYTVLYNSTSFSTFKCYGVSPSKLIDL 381
SARSr-CoV_Rs672   VFNASRFPNVYAWERTKISDCVADYTVLYNSTSFSTFKCYGVSPSKLIDL 381
6VXX              VFNATRFASVYAWNRKRISNCVADYSVLYNSASFSTFKCYGVSPTKLNDL 303
6VYB              VFNATRFASVYAWNRKRISNCVADYSVLYNSASFSTFKCYGVSPTKLNDL 295
2AJF              VFNATKFPSVYAWERKKISNCVADYSVLYNSTFFSTFKCYGVSATKL--- 52
6LZG              VFNATRFASVYAWNRKRISNCVADYSVLYNSASFSTFKCYGVSPTKLNDL 58
                  ****: *..****:*.:**:*:***:*:***: **********.:**

SARS-CoV-2        CFTNVYADSFVIRGDEVRQIAPGQTGKIADYNYKLPDDFTGCVIAWNSNN 440
SARSr-CoV_RaTG13  CFTNVYADSFVITGDEVRQIAPGQTGKIADYNYKLPDDFTGCVIAWNSKH 440
SARS-CoV_Urbani   CFSNVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFMGCVLAWNTRN 427
SARS-CoV_CUHK-W1  CFSNVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFMGCVLAWNTRN 427
SARS-CoV_GZ02     CFSNVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFMGCVLAWNTRN 427
SARS-CoV_A031     CFSNVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFMGCVLAWNTRN 427
SARS-CoV_A022     CFSNVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFMGCVLAWNTRN 427
WIV16             CFSNVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFTGCVLAWNTRN 427
WIV1              CFSNVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFTGCVLAWNTRN 428
SARSr-CoV_ZXC21   CFTSVYADTFLIRFSEVRQVAPGQTGVIADYNYKLPDDFTGCVIAWNTAK 435
SARSr-CoV_ZC45    CFTSVYADTFLIRFSEVRQVAPGQTGVIADYNYKLPDDFTGCVIAWNTAK 436
SARSr-CoV_Rp3     CFTSVYADTFLIRSSEVRQVAPGETGVIADYNYKLPDDFTGCVIAWNTAK 431
SARSr-CoV_Rs672   CFTSVYADTFLIRSSEVRQVAPGETGVIADYNYKLPDDFTGCVIAWNTAK 431
6VXX              CFTNVYADSFVIRGDEVRQIAPGQTGKIADYNYKLPDDFTGCVIAWNSNN 353
6VYB              CFTNVYADSFVIRGDEVRQIAPGQTGKIADYNYKLPDDFTGCVIAWNSNN 345
2AJF              ---NVYADSFVVKGDDVRQIAPGQTGVIADYNYKLPDDFMGCVLAWNTRN 99
6LZG              CFTNVYADSFVIRGDEVRQIAPGQTGKIADYNYKLPDDFTGCVIAWNSNN 108
                     .****:*::  .:***:***:** ************ ***:***: :

SARS-CoV-2        LDSKVGGNYNYLYRLFRKSNLKPFERDISTEIYQAGSTPCNGVEGFNCYF 490
SARSr-CoV_RaTG13  IDAKEGGNFNYLYRLFRKANLKPFERDISTEIYQAGSKPCNGQTGLNCYY 490
SARS-CoV_Urbani   IDATSTGNYNYKYRYLRHGKLRPFERDISNVPFSPDGKPCTP-PALNCYW 476
SARS-CoV_CUHK-W1  IDATSTGNYNYKYRYLRHGKLRPFERDISNVPFSPDGKPCTP-PALNCYW 476
```

```
SARS-CoV_GZ02     IDATSTGNYNYKYRYLRHGKLRPFERDISNVPFSPDGKPCTP-PALNCYW 476
SARS-CoV_A031     IDATSTGNYNYKYRYLRHGKLRPFERDISNVPFSSDGKPCTP-PAPNCYW 476
SARS-CoV_A022     IDATSTGNYNYKYRYLRHGKLRPFERDISNVPFSSDGKPCTP-PAPNCYW 476
WIV16             IDATQTGNYNYKYRSLRHGKLRPFERDISNVPFSPDGKPCTP-PAFNCYW 476
WIV1              IDATQTGNYNYKYRSLRHGKLRPFERDISNVPFSPDGKPCTP-PAFNCYW 477
SARSr-CoV_ZXC21   QDTG-----HYFYRSHRSTKLKPFERDLSSDE--------------NGVR 466
SARSr-CoV_ZC45    QDVG-----NYFYRSHRSTKLKPFERDLSSDE--------------NGVR 467
SARSr-CoV_Rp3     QDQG-----QYYYRSHRKTKLKPFERDLSSDE--------------NGVR 462
SARSr-CoV_Rs672   QDQG-----QYYYRSSRKTKLKPFERDLTSDE--------------NGVR 462
6VXX              LDS--KGNYNYLYR-------KPFERDIY--------------------F 374
6VYB              LD-----NYNYLYRLFRKSNLKPFERDIST-------------------F 371
2AJF              IDATSTGNYNYKYRYLRHGKLRPFERDISNVPFSPDGKPCTP-PALNCYW 148
6LZG              LDSKVGGNYNYLYRLFRKSNLKPFERDISTEIYQAGSTPCNGVEGFNCYF 158
                   *       :* **        :*****:

SARS-CoV-2        PLQSYGFQPTNGVGYQPYRVVVLSFELLHAPATVCGPKKSTNLVKNKCVN 540
SARSr-CoV_RaTG13  PLYRYGFYPTDGVGHQPYRVVVLSFELLNAPATVCGPKKSTNLVKNKCVN 540
SARS-CoV_Urbani   PLNDYGFYTTTGIGYQPYRVVVLSFELLNAPATVCGPKLSTDLIKNQCVN 526
SARS-CoV_CUHK-W1  PLNDYGFYTTTGIGYQPYRVVVLSFELLNAPATVCGPKLSTDLIKNQCVN 526
SARS-CoV_GZ02     PLNDYGFYTTTGIGYQPYRVVVLSFELLNAPATVCGPKLSTDLIKNQCVN 526
SARS-CoV_A031     PLRGYGFYTTSGIGYQPYRVVVLSFELLNAPATVCGPKLSTDLIKNQCVN 526
SARS-CoV_A022     PLRGYGFYTTSGIGYQPYRVVVLSFELLNAPATVCGPKLSTDLIKNQCVN 526
WIV16             PLNDYGFYITNGIGYQPYRVVVLSFELLNAPATVCGPKLSTDLIKNQCVN 526
WIV1              PLNDYGFYITNGIGYQPYRVVVLSFELLNAPATVCGPKLSTDLIKNQCVN 527
SARSr-CoV_ZXC21   TLSTYDFNPNVPLEYQATRVVVLSFELLNAPATVCGPKLSTQLVKNQCVN 516
SARSr-CoV_ZC45    TLSTYDFNPNVPLEYQATRVVVLSFELLNAPATVCGPKLSTQLVKNQCVN 517
SARSr-CoV_Rp3     TLSTYDFYPSVPVAYQATRVVVLSFELLNAPATVCGPKLSTQLVKNQCVN 512
SARSr-CoV_Rs672   TLSTYDFYPNVPIEYQATRVVVLSFELLNAPATVCGPKLSTGLVKNQCVN 512
6VXX              PLQSYGFQPT-NVGYQPYRVVVLSFELLHAPATVCGPKKSTNLVKNKCVN 423
6VYB              PLQSYGFQPT-NVGYQPYRVVVLSFELLHAPATVCGPKKSTNLVKNKCVN 420
2AJF              PLNDYGFYTTTGIGYQPYRVVVLSFE------------------------ 174
6LZG              PLQSYGFQPTNGVGYQPYRVVVLSFELLHAPATVCGP------------- 195
                  .*  *.*  .  : :*. ********
```

```
SARS-CoV-2         FNFNGLTGTGVLTESNKKFLPFQQFGRDIADTTDAVRDPQTLEILDITPC 590
SARSr-CoV_RaTG13   FNFNGLTGTGVLTESNKKFLPFQQFGRDIADTTDAVRDPQTLEILDITPC 590
SARS-CoV_Urbani    FNFNGLTGTGVLTPSSKRFQPFQQFGRDVSDFTDSVRDPKTSEILDISPC 576
SARS-CoV_CUHK-W1   FNFNGLTGTGVLTPSSKRFQPFQQFGRDVSDFTDSVRDPKTSEILDISPC 576
SARS-CoV_GZ02      FNFNGLTGTGVLTPSSKRFQPFQQFGRDVSDFTDSVRDPKTSEILDISPC 576
SARS-CoV_A031      FNFNGLTGTGVLTPSSKRFQPFQQFGRDVSDFTDSVRDPKTSEILDISPC 576
SARS-CoV_A022      FNFNGLTGTGVLTPSSKRFQPFQQFGRDVSDFTDSVRDPKTSEILDISPC 576
WIV16              FNFNGLTGTGVLTPSSKRFQPFQQFGRDVLDFTDSVRDPKTSEILDISPC 576
WIV1               FNFNGLTGTGVLTPSSKRFQPFQQFGRDVSDFTDSVRDPKTSEILDISPC 577
SARSr-CoV_ZXC21    FNFNGLKGTGVLTDSSKRFQSFQQFGKDASDFIDSVRDPQTLEILDITPC 566
SARSr-CoV_ZC45     FNFNGLKGTGVLTDSSKRFQSFQQFGKDASDFIDSVRDPQTLEILDITPC 567
SARSr-CoV_Rp3      FNFNGLKGTGVLTESSKRFQSFQQFGRDTSDFTDSVRDPQTLEILDISPC 562
SARSr-CoV_Rs672    FNFNGLKGTGVLTDSSKRFQSFQQFGRDTSDFTDSVRDPQTLQILDITPC 562
6VXX               FNFNGLTGTGVLTESNKKFLPFQQFGRDIADTTDAVRDPQTLEILDITPC 473
6VYB               FNFNGLTGTGVLTESNKKFLPFQQFGRDIADTTDAVRDPQTLEILDITPC 470
2AJF               -------------------------------------------------- 174
6LZG               -------------------------------------------------- 195


SARS-CoV-2         SFGGVSVITPGTNTSNQVAVLYQDVNCTEVPVAIHADQLTPTWRVYSTGS 640
SARSr-CoV_RaTG13   SFGGVSVITPGTNASNQVAVLYQDVNCTEVPVAIHADQLTPTWRVYSTGS 640
SARS-CoV_Urbani    SFGGVSVITPGTNASSEVAVLYQDVNCTDVSTAIHADQLTPAWRIYSTGN 626
SARS-CoV_CUHK-W1   SFGGVSVITPGTNASSEVAVLYQDVNCTDVSTAIHADQLTPAWRIYSTGN 626
SARS-CoV_GZ02      SFGGVSVITPGTNASSEVAVLYQDVNCTDVSTAIHADQLTPAWRIYSTGN 626
SARS-CoV_A031      SFGGVSVITPGTNASSEVAVLYQDVNCTDVSTLIHAEQLTPAWRIYSTGN 626
SARS-CoV_A022      SFGGVSVITPGTNASSEVAVLYQDVNCTDVSTLIHAEQLTPAWRIYSTGN 626
WIV16              SFGGVSVITPGTNTSSEVAVLYQDVNCTDVPVAIHADQLTPSWRVYSTGN 626
WIV1               SFGGVSVITPGTNTSSEVAVLYQDVNCTDVPVAIHADQLTPSWRVHSTGN 627
SARSr-CoV_ZXC21    SFGGVSVITPGTNTSSEVAVLYQDVNCTDVPTTIHADQLTPAWRIYAIGT 616
SARSr-CoV_ZC45     SFGGVSVITPGTNTSLEVAVLYQDVNCTDVPTTIHADQLTPAWRIYATGT 617
SARSr-CoV_Rp3      SFGGVSVITPGTNASSEVAVLYQDVNCTDVPAAIHADQLTPAWRVYSTGT 612
SARSr-CoV_Rs672    SFGGVSVITPGTNASSEVAVLYQDVNCTDVPTAIRADQLTPAWRVYSTGV 612
```

```
6VXX              SFGGVSVITPGTNTSNQVAVLYQDVNCTEV-------------------- 503
6VYB              SFGGVSVITPGTNTSNEVAVLYQDVNCTEV-------------------- 500
2AJF              -------------------------------------------------- 174
6LZG              -------------------------------------------------- 195


SARS-CoV-2        NVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQTQTNSPRRARSVASQ 690
SARSr-CoV_RaTG13  NVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQTQT----NSRSVASQ 686
SARS-CoV_Urbani   NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS----LLRSTSQK 672
SARS-CoV_CUHK-W1  NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS----LLRSTSQK 672
SARS-CoV_GZ02     NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS----LLRSTSQK 672
SARS-CoV_A031     NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS----SLRSTSQK 672
SARS-CoV_A022     NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS----SLRSTSQK 672
WIV16             NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS----SLRSTSQK 672
WIV1              NVFQTQAGCLIGAEHVDTSYECDIPIGAGICASYHTVS----SLRSTSQK 673
SARSr-CoV_ZXC21   SVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTAS----ILRSTGQK 662
SARSr-CoV_ZC45    NVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTAS----ILRSTSQK 663
SARSr-CoV_Rp3     NVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTAS----TLRSVGQK 658
SARSr-CoV_Rs672   NVFQTQAGCLIGAEHVNASYECDIPIGAGICASYHTAS----TLRSVGQK 658
6VXX              NVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQT------------SQ 541
6VYB              NVFQTRAGCLIGAEHVNNSYECDIPIGAGICASYQT-------------Q 537
2AJF              -------------------------------------------------- 174
6LZG              -------------------------------------------------- 195


SARS-CoV-2        SIIAYTMSLGAENSVAYSNNSIAIPTNFTISVTTEILPVSMTKTSVDCTM 740
SARSr-CoV_RaTG13  SIIAYTMSLGAENSVAYSNNSIAIPTNFTISVTTEILPVSMTKTSVDCTM 736
SARS-CoV_Urbani   SIVAYTMSLGADSSIAYSNNTIAIPTNFSISITTEVMPVSMAKTSVDCNM 722
SARS-CoV_CUHK-W1  SIVAYTMSLGADSSIAYSNNTIAIPTNFSISITTEVMPVSMAKTSVDCNM 722
SARS-CoV_GZ02     SIVAYTMSLGADSSIAYSNNTIAIPTNFSISITTEVMPVSMAKTSVDCNM 722
SARS-CoV_A031     SIVAYTMSLGADSSIAYSNNTIAIPTNFSISITTEVMPVSMAKTSVDCNM 722
SARS-CoV_A022     SIVAYTMSLGADSSIAYSNNTIAIPTNFSISITTEVMPVSMAKTSVDCNM 722
WIV16             SIVAYTMSLGADSSIAYSNNTIAIPTNFSISITTEVMPVSMAKTSVDCNM 722
```

```
WIV1               SIVAYTMSLGADSSIAYSNNTIAIPTNFSISITTEVMPVSMAKTSVDCNM 723
SARSr-CoV_ZXC21    AIVAYTMSLGAENSIAYANNSIAIPTNFSISVTTEVMPVSMAKTSVDCTM 712
SARSr-CoV_ZC45     AIVAYTMSLGAENSIAYANNSIAIPTNFSISVTTEVMPVSMAKTSVDCTM 713
SARSr-CoV_Rp3      SIVAYTMSLGAENSIAYANNSIAIPTNFSISVTTEVMPVSMAKTSVDCTM 708
SARSr-CoV_Rs672    SIVAYTMSLGAENSIAYANNSIAIPTNFSISVTTEVMPVSMAKTSVDCTM 708
6VXX               SIIAYTMSLGAENSVAYSNNSIAIPTNFTISVTTEILPVSMTKTSVDCTM 591
6VYB               SIIAYTMSLGAENSVAYSNNSIAIPTNFTISVTTEILPVSMTKTSVDCTM 587
2AJF               -------------------------------------------------- 174
6LZG               -------------------------------------------------- 195


SARS-CoV-2         YICGDSTECSNLLLQYGSFCTQLNRALTGIAVEQDKNTQEVFAQVKQIYK 790
SARSr-CoV_RaTG13   YICGDSTECSNLLLQYGSFCTQLNRALTGIAVEQDKNTQEVFAQVKQIYK 786
SARS-CoV_Urbani    YICGDSTECANLLLQYGSFCTQLNRALSGIAAEQDRNTREVFAQVKQMYK 772
SARS-CoV_CUHK-W1   YICGDSTECANLLLQYGSFCTQLNRALSGIAAEQDRNTREVFAQVKQMYK 772
SARS-CoV_GZ02      YICGDSTECANLLLQYGSFCTQLNRALSGIAAEQDRNTREVFAQVKQMYK 772
SARS-CoV_A031      YICGDSTECANLLLQYGSFCRQLNRALSGIAAEQDRNTREVFVQVKQMYK 772
SARS-CoV_A022      YICGDSTECANLLLQYGSFCRQLNRALSGIAAEQDRNTREVFVQVKQMYK 772
WIV16              YICGDSTECANLLLQYGSFCTQLNRALSGIAVEQDRNTREVFAQVKQMYK 772
WIV1               YICGDSTECANLLLQYGSFCTQLNRALSGIAVEQDRNTREVFAQVKQMYK 773
SARSr-CoV_ZXC21    YICGDSIECSNLLLQYGSFCTQLNRALSGIAIEQDKNTQEVFAQVKQIYK 762
SARSr-CoV_ZC45     YICGDSIECSNLLLQYGSFCTQLNRALSGIAIEQDKNTQEVFAQVKQIYK 763
SARSr-CoV_Rp3      YICGDSLECSNLLLQYGSFCTQLNRALSGIAIEQDKNTQEVFAQVKQMYK 758
SARSr-CoV_Rs672    YICGDSQECSNLLLQYGSFCTQLNRALTGVALEQDKNTQEVFAQVKQMYK 758
6VXX               YICGDSTECSNLLLQYGSFCTQLNRALTGIAVEQDKNTQEVFAQVKQIYK 641
6VYB               YICGDSTECSNLLLQYGSFCTQLNRALTGIAVEQDKNTQEVFAQVKQIYK 637
2AJF               -------------------------------------------------- 174
6LZG               -------------------------------------------------- 195


SARS-CoV-2         TPPIKDFGGFNFSQILPDPSKPSKRSFIEDLLFNKVTLADAGFIKQYGDC 840
SARSr-CoV_RaTG13   TPPIKDFGGFNFSQILPDPSKPSKRSFIEDLLFNKVTLADAGFIKQYGDC 836
SARS-CoV_Urbani    TPTLKYFGGFNFSQILPDPLKPTKRSFIEDLLFNKVTLADAGFMKQYGEC 822
```

```
SARS-CoV_CUHK-W1  TPTLKYFGGFNFSQILPDPLKPTKRSFIEDLLFNKVTLADAGFMKQYGEC 822
SARS-CoV_GZ02     TPTLKDFGGFNFSQILPDPLKPTKRSFIEDLLFNKVTLADAGFMKQYGEC 822
SARS-CoV_A031     TPTLKDFGGFNFSQILPDPLKPTKRSFIEDLLFNKVTLADAGFMKQYGQC 822
SARS-CoV_A022     TPTLKDFGGFNFSQILPDPLKPTKRSFIEDLLFNKVTLADAGFMKQYGQC 822
WIV16             TPTLKDFGGFNFSQILPDPLKPTKRSFIEDLLFNKVTLADAGFMKQYGEC 822
WIV1              TPTLKDFGGFNFSQILPDPLKPTKRSFIEDLLFNKVTLADAGFMKQYGEC 823
SARSr-CoV_ZXC21   TPPIKDFGGFNFSQILPDPSKPSKRSFIEDLLFNKVTLADAGFIKQYGDC 812
SARSr-CoV_ZC45    TPPIKDFGGFNFSQILPDPSKPSKRSFIEDLLFNKVTLADAGFIKQYGDC 813
SARSr-CoV_Rp3     TPAIKDFGGFNFSQILPDPSKPTKRSFIEDLLFNKVTLADAGFMKQYGEC 808
SARSr-CoV_Rs672   TPAIKDFGGFNFSQILPDPSKPTKRSFIEDLLFNKVTLADAGFMKQYGEC 808
6VXX              TPPIKDFGGFNFSQILPDPSKPSKRSFIEDLLFNKVT------------- 678
6VYB              TPPIKDFGGFNFSQILPDPSK-SKRSFIEDLLFNKVT------------- 673
2AJF              -------------------------------------------------- 174
6LZG              -------------------------------------------------- 195


SARS-CoV-2        LGDIAARDLICAQKFNGLTVLPPLLTDEMIAQYTSALLAGTITSGWTFGA 890
SARSr-CoV_RaTG13  LGDIAARDLICAQKFNGLTVLPPLLTDEMIAQYTSALLAGTITSGWTFGA 886
SARS-CoV_Urbani   LGDINARDLICAQKFNGLTVLPPLLTDDMIAAYTAALVSGTATAGWTFGA 872
SARS-CoV_CUHK-W1  LGDINARDLICAQKFNGLTVLPPLLTDDMIAAYTAALVSGTATAGWTFGA 872
SARS-CoV_GZ02     LGDINARDLICAQKFNGLTVLPPLLTDDMIAAYTAALVSGTATAGWTFGA 872
SARS-CoV_A031     LGDINARDLICAQKFNGLTVLPPLLTDDMIAAYTAALVSGTATAGWTFGA 872
SARS-CoV_A022     LGDINARDLICAQKFNGLTVLPPLLTDDMIAAYTAALVSGTATAGWTFGA 872
WIV16             LGDINARDLICAQKFNGLTVLPPLLTDDMIAAYTAALVSGTATAGWTFGA 872
WIV1              LGDINARDLICAQKFNGLTVLPPLLTDDMIAAYTAALVSGTATAGWTFGA 873
SARSr-CoV_ZXC21   LGDISARDLICAQKFNGLTVLPPLLTDEMIAAYTAALISGTATAGWTFGA 862
SARSr-CoV_ZC45    LGGISARDLICAQKFNGLTVLPPLLTDEMIAAYTAALISGTATAGWTFGA 863
SARSr-CoV_Rp3     LGDISARDLICAQKFNGLTVLPPLLTDEMIAAYTAALVSGTATAGWTFGA 858
SARSr-CoV_Rs672   LGDISARDLICAQKFNGLTVLPPLLTDEMIAAYTAALVSGTATAGWTFGA 858
6VXX              -------------KFNGLTVLPPLLTDEMIAQYTSALLAGTITSGWTFGA 715
6VYB              --------------FNGLTVLPPLLTDEMIAQYTSALLAGTITSGWTFGA 709
2AJF              -------------------------------------------------- 174
6LZG              -------------------------------------------------- 195
```

```
SARS-CoV-2         GAALQIPFAMQMAYRFNGIGVTQNVLYENQKLIANQFNSAIGKIQDSLSS 940
SARSr-CoV_RaTG13   GAALQIPFAMQMAYRFNGIGVTQNVLYENQKLIANQFNSAIGKIQDSLSS 936
SARS-CoV_Urbani    GAALQIPFAMQMAYRFNGIGVTQNVLYENQKQIANQFNKAISQIQESLTT 922
SARS-CoV_CUHK-W1   GAALQIPFAMQMAYRFNGIGVTQNVLYENQKQIANQFNKAISQIQESLTT 922
SARS-CoV_GZ02      GAALQIPFAMQMAYRFNGIGVTQNVLYENQKQIANQFNKAISQIQESLTT 922
SARS-CoV_A031      GAALQIPFAMQMAYRFNGIGVTQNVLYENQKQIANQFNKAISQIQESLTT 922
SARS-CoV_A022      GAALQIPFAMQMAYRFNGIGVTQNVLYENQKQIANQFNKAISQIQESLTT 922
WIV16              GAALQIPFAMQMAYRFNGIGVTQNVLYENQKQIANQFNKAISQIQESLTT 922
WIV1               GAALQIPFAMQMAYRFNGIGVTQNVLYENQKQIANQFNKAISQIQESLTT 923
SARSr-CoV_ZXC21    GAALQIPFAMQMAYRFNGIGVTQNVLYENQKLIANQFNSAIGKIQESLTS 912
SARSr-CoV_ZC45     GAALQIPFAMQMAYRFNGIGVTQNVLYENQKLIANQFNSAIGKIQESLTS 913
SARSr-CoV_Rp3      GSALQIPFAMQMAYRFNGIGVTQNVLYENQKQIANQFNKAISQIQESLTT 908
SARSr-CoV_Rs672    GAALQIPFAMQMAYRFNGIGVTQNVLYENQKQIANQFNKAISQIQESLTT 908
6VXX               GAALQIPFAMQMAYRFNGIGVTQNVLYENQKLIANQFNSAIGKIQDSLSS 765
6VYB               GAALQIPFAMQMAYRFNGIGVTQNVLYENQKLIANQFNSAIGKIQDSLSS 759
2AJF               -------------------------------------------------- 174
6LZG               -------------------------------------------------- 195
```

```
SARS-CoV-2         TASALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 990
SARSr-CoV_RaTG13   TASALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 986
SARS-CoV_Urbani    TSTALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 972
SARS-CoV_CUHK-W1   TSTALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 972
SARS-CoV_GZ02      TSTALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 972
SARS-CoV_A031      TSTALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 972
SARS-CoV_A022      TSTALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 972
WIV16              TSTALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 972
WIV1               TSTALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 973
SARSr-CoV_ZXC21    TASALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 962
SARSr-CoV_ZC45     TASALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 963
SARSr-CoV_Rp3      TSTALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 958
```

```
SARSr-CoV_Rs672   TSTALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDKVEAE 958
6VXX              TASALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDPPEAE 815
6VYB              TASALGKLQDVVNQNAQALNTLVKQLSSNFGAISSVLNDILSRLDPPEAE 809
2AJF              -------------------------------------------------- 174
6LZG              -------------------------------------------------- 195


SARS-CoV-2        VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1040
SARSr-CoV_RaTG13  VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1036
SARS-CoV_Urbani   VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1022
SARS-CoV_CUHK-W1  VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1022
SARS-CoV_GZ02     VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1022
SARS-CoV_A031     VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1022
SARS-CoV_A022     VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1022
WIV16             VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1022
WIV1              VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1023
SARSr-CoV_ZXC21   VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1012
SARSr-CoV_ZC45    VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1013
SARSr-CoV_Rp3     VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 1008
SARSr-CoV_Rs672   VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVPGQSKRV 1008
6VXX              VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 865
6VYB              VQIDRLITGRLQSLQTYVTQQLIRAAEIRASANLAATKMSECVLGQSKRV 859
2AJF              -------------------------------------------------- 174
6LZG              -------------------------------------------------- 195


SARS-CoV-2        DFCGKGYHLMSFPQSAPHGVVFLHVTYVPAQEKNFTTAPAICHDGKAHFP 1090
SARSr-CoV_RaTG13  DFCGKGYHLMSFPQSAPHGVVFLHVTYVPAQEKNFTTAPAICHDGKAHFP 1086
SARS-CoV_Urbani   DFCGKGYHLMSFPQAAPHGVVFLHVTYVPSQERNFTTAPAICHEGKAYFP 1072
SARS-CoV_CUHK-W1  DFCGKGYHLMSFPQAAPHGVVFLHVTYVPSQERNFTTAPAICHEGKAYFP 1072
SARS-CoV_GZ02     DFCGKGYHLMSFPQAAPHGVVFLHVTYVPSQERNFTTAPAICHEGKAYFP 1072
SARS-CoV_A031     DFCGKGYHLMSFPQAAPHGVVFLHVTYVPSQERNFTTAPAICHEGKAYFP 1072
SARS-CoV_A022     DFCGKGYHLMSFPQAAPHGVVFLHVTYVPSQERNFTTAPAICHEGKAYFP 1072
```

```
WIV16              DFCGKGYHLMSFPQAAPHGVVFLHVTYVPSQERNFTTAPAICHEGKAYFP 1072
WIV1               DFCGKGYHLMSFPQAAPHGVVFLHVTYVPSQERNFTTAPAICHEGKAYFP 1073
SARSr-CoV_ZXC21    DFCGKGYHLMSFPQSAPHGVVFLHVTYIPSQEKNFTTAPAICHEGKAHFP 1062
SARSr-CoV_ZC45     DFCGKGYHLMSFPQSAPHGVVFLHVTYIPSQEKNFTTAPAICHEGKAHFP 1063
SARSr-CoV_Rp3      DFCGKGYHLMSFPQAAPHGVVFLHVTYVPSQERNFTTAPAICHEGKAYFP 1058
SARSr-CoV_Rs672    DFCGRGYHLMSFPQAAPHGVVFLHVTYVPSQEKNFTTAPAICHEGKAYFP 1058
6VXX               DFCGKGYHLMSFPQSAPHGVVFLHVTYVPAQEKNFTTAPAICHDGKAHFP 915
6VYB               DFCGKGYHLMSFPQSAPHGVVFLHVTYVPAQEKNFTTAPAICHDGKAHFP 909
2AJF               -------------------------------------------------- 174
6LZG               -------------------------------------------------- 195


SARS-CoV-2         REGVFVSNGTHWFVTQRNFYEPQIITTDNTFVSGNCDVVIGIVNNTVYDP 1140
SARSr-CoV_RaTG13   REGVFVSNGTHWFVTQRNFYEPQIITTDNTFVSGSCDVVIGIVNNTVYDP 1136
SARS-CoV_Urbani    REGVFVFNGTSWFITQRNFFSPQIITTDNTFVSGNCDVVIGIINNTVYDP 1122
SARS-CoV_CUHK-W1   REGVFVFNGTSWFITQRNFFSPQIITTDNTFVSGNCDVVIGIINNTVYDP 1122
SARS-CoV_GZ02      REGVFVFNGTSWFITQRNFFSPQIITTDNTFVSGNCDVVIGIINNTVYDP 1122
SARS-CoV_A031      REGVFVFSGTSWFITQRNFFSPQIITTDNTFVSGNCDVVIGIINNTVYDP 1122
SARS-CoV_A022      REGVFVFSGTSWFITQRNFFSPQIITTDNTFVSGNCDVVIGIINNTVYDP 1122
WIV16              REGVFVFNGTSWFITQRNFFSPQIITTDNTFVSGSCDVVIGIINNTVYDP 1122
WIV1               REGVFVFNGTSWFITQRNFFSPQIITTDNTFVSGSCDVVIGIINNTVYDP 1123
SARSr-CoV_ZXC21    REGVFVSNGTHWFVTQRNFYEPQIITTDNTFVSGNCDVVIGIINNTVYDP 1112
SARSr-CoV_ZC45     REGVFVSNGTHWFVTQRNFYEPKIITTDNTFVSGNCDVVIGIINNTVYDP 1113
SARSr-CoV_Rp3      REGVFVSNGTSWFITQRNFYSPQIITTDNTFVAGSCDVVIGIINNTVYDP 1108
SARSr-CoV_Rs672    REGVFVSNGTSWFITQRNFYSPQIITTDNTFVAGNCDVVIGIINNTVYDP 1108
6VXX               REGVFVSNGTHWFVTQRNFYEPQIITTDNTFVSGNCDVVIGIVNNTVYDP 965
6VYB               REGVFVSNGTHWFVTQRNFYEPQIITTDNTFVSGNCDVVIGIVNNTVYDP 959
2AJF               -------------------------------------------------- 174
6LZG               -------------------------------------------------- 195


SARS-CoV-2         LQPELDSFKEELDKYFKNHTSPDVDLGDISGINASVVNIQKEIDRLNEVA 1190
SARSr-CoV_RaTG13   LQPELDSFKEELDKYFKNHTSPDVDLGDISGINASVVNIQKEIDRLNEVA 1186
```

```
SARS-CoV_Urbani    LQPELDSFKEELDKYFKNHTSPDVDLGDISGINASVVNIQKEIDRLNEVA 1172
SARS-CoV_CUHK-W1   LQPELDSFKEELDKYFKNHTSPDVDLGDISGINASVVNIQKEIDRLNEVA 1172
SARS-CoV_GZ02      LQPELDSFKEELDKYFKNHTSPDVDLGDISGINASVVNIQEEIDRLNEVA 1172
SARS-CoV_A031      LQPELDSFKEELDKYFKNHTSPDVDLGDISGINASVVNIQEEIDRLNEVA 1172
SARS-CoV_A022      LQPELDSFKEELDKYFKNHTSPDVDLGDISGINASVVNIQEEIDRLNEVA 1172
WIV16              LQPELDSFKEELDKYFKNHTSPDVDLGDISGINASVVNIQKEIDRLNEVA 1172
WIV1               LQPELDSFKEELDKYFKNHTSPDVDLGDISGINASVVNIQKEIDRLNEVA 1173
SARSr-CoV_ZXC21    LQPELDSFKEELDKYFKNHTSPDIDLGDISGINASVVNIQKEIDRLNEVA 1162
SARSr-CoV_ZC45     LQPELDSFKEELDKYFKNHTSPDIDLGDISGINASVVNIQKEIDRLNEVA 1163
SARSr-CoV_Rp3      LQPELDSFKEELDKYFKNHTSPDVDLGDISGINASVVNIQKEIDRLNEVA 1158
SARSr-CoV_Rs672    LQPELDSFKEELDKYFKNHTSPDVDLGDISGINASVVNIQKEIDRLNEVA 1158
6VXX               LQPELDS------------------------------------------- 972
6VYB               LQPELDS------------------------------------------- 966
2AJF               -------------------------------------------------- 174
6LZG               -------------------------------------------------- 195


SARS-CoV-2         KNLNESLIDLQELGKYEQYIKWPWYIWLGFIAGLIAIVMVTIMLCCMTSC 1240
SARSr-CoV_RaTG13   KNLNESLIDLQELGKYEQYIKWPWYIWLGFIAGLIAIIMVTIMLCCMTSC 1236
SARS-CoV_Urbani    KNLNESLIDLQELGKYEQYIKWPWYVWLGFIAGLIAIVMVTILLCCMTSC 1222
SARS-CoV_CUHK-W1   KNLNESLIDLQELGKYEQYIKWPWYVWLGFIAGLIAIVMVTILLCCMTSC 1222
SARS-CoV_GZ02      KNLNESLIDLQELGKYEQYIKWPWYVWLGFIAGLIAIVMVTILLCCMTSC 1222
SARS-CoV_A031      KNLNESLIDLQELGKYEQYIKWPWYVWLGFIAGLIAIVMVTILLCCMTSC 1222
SARS-CoV_A022      KNLNESLIDLQELGKYEQYIKWPWYVWLGFIAGLIAIVMVTILLCCMTSC 1222
WIV16              KNLNESLIDLQELGKYEQYIKWPWYVWLGFIAGLIAIVMVTILLCCMTSC 1222
WIV1               KNLNESLIDLQELGKYEQYIKWPWYVWLGFIAGLIAIVMVTILLCCMTSC 1223
SARSr-CoV_ZXC21    RNLNESLIDLQELGKYEHYIKWPWYVWLGFIAGLIAIVMVTILLCCMTSC 1212
SARSr-CoV_ZC45     RNLNESLIDLQELGKYEQYIKWPWYVWLGFIAGLIAIVMVTILLCCMTSC 1213
SARSr-CoV_Rp3      KNLNESLIDLQELGKYEQYIKWPWYVWLGFIAGLIAIVMVTILLCCMTSC 1208
SARSr-CoV_Rs672    KNLNESLIDLQELGKYEQYIKWPWYVWLGFIAGLIAIVMATILLCCMTSC 1208
6VXX               -------------------------------------------------- 972
6VYB               -------------------------------------------------- 966
2AJF               -------------------------------------------------- 174
```

```
6LZG              -------------------------------------------------- 195


SARS-CoV-2        CSCLKGCCSCGSCCKFDEDDSEPVLKGVKLHYT 1273
SARSr-CoV_RaTG13  CSCLKGCCSCGSCCKFDEDDSEPVLKGVKLHYT 1269
SARS-CoV_Urbani   CSCLKGACSCGSCCKFDEDDSEPVLKGVKLHYT 1255
SARS-CoV_CUHK-W1  CSCLKGACSCGSCCKFDEDDSEPVLKGVKLHYT 1255
SARS-CoV_GZ02     CSCLKGACSCGSCCKFDEDDSEPVLKGVKLHYT 1255
SARS-CoV_A031     CSCLKGACSCGSCCKFDEDDSEPVLKGVKLHYT 1255
SARS-CoV_A022     CSCLKGACSCGSCCKFDEDDSEPVLKGVKLHYT 1255
WIV16             CSCLKGACSCGSCCKFDEDDSEPVLKGVKLHYT 1255
WIV1              CSCLKGACSCGSCCKFDEDDSEPVLKGVKLHYT 1256
SARSr-CoV_ZXC21   CSCLKGCCSCGFCCKFDEDDSEPVLKGVKLHYT 1245
SARSr-CoV_ZC45    CSCLKGCCSCGSCCKFDEDDSEPVLKGVKLHYT 1246
SARSr-CoV_Rp3     CSCLKGACSCGSCCKFDEDDSEPVLKGVKLHYT 1241
SARSr-CoV_Rs672   CSCLKGACSCGSCCKFDEDDSEPVLKGVKLHYT 1241
6VXX              --------------------------------- 972
6VYB              --------------------------------- 966
2AJF              --------------------------------- 174
6LZG              --------------------------------- 195
```

Appendix B

PyMOL scripts

```
# Fetch chain A only of PDB 6VYB
fetch 6VYBA
as cartoon
hide sticks
color lightblue
color tv_red, resi 673-693
set label_size, 30
set label_position =[-1, 1, 8]
set label_color, black
label /6VYBA/A/A/THR`676/CA, "THR 676"
label /6VYBA/A/A/GLN`690/CA, "GLN 690"
bg_color white

set_view (\
     0.286217600,   -0.011345523,   -0.958088458,\
    -0.948126435,    0.140928283,   -0.284912586,\
     0.138257042,    0.989944160,    0.029574998,\
     0.003286857,   -0.003728534,  -50.905109406,\
   174.548110962,  232.581466675,  187.326797485,\
   -69.996986389,  173.154022217,  -20.000000000 )
```

```
set ray_trace_mode, 1
png cov2-spike-6VYB.png, 1400, dpi=300, ray=1
```

Appendix C

About the Author

Jean-Yves Sgro, a senior scientist with years of experience in using and teaching computer programs, creates, organizes and teaches hands on workshops.

Jean-Yves has been at UW since 1986 after a Master in Physiology and a Ph.D. in Cellular and Molecular Biology from Joseph Fourier University, Grenoble, France, and researched at the European Molecular Biology Laboratory (EMBL) where he already used large computers for sequence analysis.

In Madison, at the Institute for Molecular Virology (IMV) he continued developing computer expertise in addition to his wet-lab research – 3D molecular visualization (virusworld), RNA-folding predictions, sequence and data analysis…

In 1996 he joined the UW Biotechnology Center to better help Campus biologists analyze and visualize their data while continuing research at IMV until 2014 when this part-time position was transferred to the Biochemistry Department where he organizes and teaches hands-on tutorials on molecular graphics, data analysis as a support to the department personnel.

Tutorials are available on line from the The Biochemistry Computational Research Facility (BCRF.)

Bibliography

Lokman, S. M., Rasheduzzaman, M., Salauddin, A., Barua, R., Tanzina, A. Y., Rumi, M. H., Hossain, M. I., Siddiki, A. M. A. M. Z., Mannan, A., and Hasan, M. M. (2020). Exploring the genomic and proteomic variations of SARS-CoV-2 spike glycoprotein: A computational biology approach. *Infect. Genet. Evol.*, 84:104389. [PubMed Central:PMC7266584] [DOI:10.1016/j.meegid.2020.104389] [PubMed:32502733].

Notredame, C., Higgins, D. G., and Heringa, J. (2000). T-Coffee: A novel method for fast and accurate multiple sequence alignment. *J. Mol. Biol.*, 302(1):205–217. [DOI:10.1006/jmbi.2000.4042] [PubMed:10964570].

Schrödinger, LLC (2020). The PyMOL molecular graphics system, version 2.0.

Sievers, F., Wilm, A., Dineen, D., Gibson, T. J., Karplus, K., Li, W., Lopez, R., McWilliam, H., Remmert, M., S?ding, J., Thompson, J. D., and Higgins, D. G. (2011). Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. *Mol. Syst. Biol.*, 7:539. [PubMed Central:PMC3261699] [DOI:10.1038/msb.2011.75] [PubMed:20639539].

Thompson, S. (2009). *An Introduction to Multiple Sequence Alignment — and the T-Coffee Shop. Beyond Just Aligning Sequences: How Good can you Make your Alignment, and so What?*, pages 283–313.

Walls, A. C., Park, Y. J., Tortorici, M. A., Wall, A., McGuire, A. T., and Veesler, D. (2020). Structure, Function, and Antigenicity of the SARS-CoV-2 Spike Glycoprotein. *Cell*, 181(2):281–292. [PubMed Central:PMC7102599] [DOI:10.1016/j.cell.2020.02.058] [PubMed:32155444].

Wang, Q., Zhang, Y., Wu, L., Niu, S., Song, C., Zhang, Z., Lu, G., Qiao, C., Hu, Y., Yuen, K. Y., Wang, Q., Zhou, H., Yan, J., and Qi, J. (2020). Structural and Functional Basis of SARS-CoV-2 Entry by Using Human ACE2. *Cell*, 181(4):894–904. [PubMed Central:PMC7144619] [DOI:10.1016/j.cell.2020.03.045] [PubMed:32275855].