02-251 COVID-19 Challenge
Evolutionary Trees/Multiple Alignment
Kunal Joshi & Phillip Compeau

Introduction

In the previous assignment, we demonstrated how to assemble and annotate the SARS-CoV-2 genome that caused the COVID-19 pandemic. Although knowing the genomic identity of this virus is very important in order to compare it to other coronaviruses (most notably the SARS-CoV virus that caused the 2003 outbreak of SARS), researchers are interested in determining the genome of thousands of different individual viruses as the virus spreads around the planet.

This idea of viral **genomic surveillance** is not new; it became famous in 2014 when researchers captured patient samples to identify the source of the Ebola outbreak.[1] In the case of SARS-CoV-2, we know that the virus likely originated in Wuhan, but we want to apply genomic surveillance to determine how the virus is mutating in human hosts as it spreads, and to identify new variants of the virus for further study, such as whether vaccines work on these variants.

The computational task at hand is to construct an evolutionary tree, along with a multiple alignment, of viral genomes sampled from many patients. We know from class that this requires sophisticated algorithms that are efficient enough to handle many 30,000 base pair genomes.

We will be studying samples taken from patients in the United Kingdom (UK), not because the virus there has an affinity for milky tea, but because their reliance on a nationalized health service means that they have collected and published more viral genome data than we have in the US.

In this assignment, we will place ourselves in the shoes of researchers sequencing genomes and see if we can find any interesting variants within the patient viral genomes. Are the variants we find changing over time?

Capturing and aligning some patient genomes

On each of six days in November and December 2020, we collected a random sample of approximately 100 coronavirus genomes in patients in England, taken from the COVID-19

---

1 Gire et al. Genomic surveillance elucidates Ebola virus origin and transmission during the 2014 outbreak. https://pubmed.ncbi.nlm.nih.gov/25214632/.

Genomics UK Consortium.[2] These files are contained within a folder of the master folder for this assignment (click here to download). We took samples at different times so that we can see how the virus genomes are changing over time at a population level.

Below are the collection dates and corresponding file names containing that particular date's sample. These files are in FASTA format, which we introduced in the assembly/annotation challenge; feel free to take a peek at each file using a text editor and see the genomes.

- November 03, 2020 (`HCOV19-ENGLAND-031120.fasta`)
- November 10, 2020 (`HCOV19-ENGLAND-101120.fasta`)
- November 17, 2020 (`HCOV19-ENGLAND-171120.fasta`)
- November 27, 2020 (`HCOV19-ENGLAND-271120.fasta`)
- December 05, 2020 (`HCOV19-ENGLAND-051220.fasta`)
- December 08, 2020 (`HCOV19-ENGLAND-081220.fasta`)

An important note is that the November 17 file is corrupted and did not align properly. We will not use it in our analysis, but we will use this as a teachable moment and encourage you to examine this file in the next section to discover went wrong.

To analyze our data, we will perform a **multiple sequence alignment** of each file, which will reveal possible mutations at each position of the genomes. In this assignment, we will use Clustal, which learned in class produces an evolutionary "guide" tree in order to produce a multiple sequence alignment. Recall that Clustal is a heuristic, meaning it may not produce the optimal alignment solution but has manageable asymptotic complexity.

The most up-to-date, optimized version of Clustal (known as Clustal Omega) still takes approximately two hours to run on 100 coronavirus genomes; this is why we took only 100 samples on each day instead of 1000. For your convenience, we have already aligned each sample: the resulting alignment for each file is stored in (`PREV_FILE_NAME-A.fasta`). For example, the aligned sample of (`HCOV19-ENGLAND-031120.fasta`) is stored within (`HCOV19-ENGLAND-031120-A.fasta`). It is outside the scope of this assignment, but if you are interested in running the alignment algorithm on each sample yourself, you can do so by using the European Bioinformatics Institute's free Clustal tool, or by running Clustal on a Galaxy server, or by downloading a program that has Clustal built into it, such as MEGA.

Finding mutations

First, rather than using a text editor, we will use a specialized tool to view a multiple sequence alignment of 100 viral genomes. The National Center for Biotechnology Information (NCBI) provides a great MSA viewer at the link below.

[2] https://www.cogconsortium.uk.

https://www.ncbi.nlm.nih.gov/projects/msaviewer/

Visit this link, and upload the November 3 file (`HCOV19-ENGLAND-031120-A.fasta`). You can do this by clicking on the `Upload` button, selecting `Data File` from the sidebar menu, and following the upload instructions. Then click `Close` when the data has been uploaded.

Once completed, the viewer should open, as shown in the screenshot below.

We cannot see individual nucleotides currently because the viewer is presenting an eagle-eye view of the alignment. You can obtain more information about a given position by clicking on it. You can also use the zoom feature at the top of the page to focus on a smaller region of the alignment.

Exercise 1: All 100 genomes have the same substring of five nucleotides ranging from column 2251 of the alignment to column 2255. What is the substring?

Answer (2 points): ACAAA.

Exercise 2: What do the red vertical bars signify in the multiple alignment? What about the gray regions? (Hint: You may find IUPAC notation helpful.)

Answer (3 points; 1 for each part, 1 for reasonable attempt): The red vertical bars correspond to nucleotides that differ from the consensus. The gray regions contain “N” nucleotides, which students can explain in one of two ways. For one, they could correspond to gaps in the alignment (they don’t). They likely correspond to regions that are between contigs in the genome being considered.

Exercise 3: Are there any regions of the genome that you find particularly interesting in terms of studying viral variation? Justify your answer.

Answer (3 points): Students can really say what they like here, but what we are looking for is that they identify a region where the genomes vary significantly, rather than isolated mutations. This can mean a single point with many mutations, or it can mean a region of the genome (e.g., spike protein) where there seems to be more variability.

Exercise 4: If you look at the ends of the alignment, you will see that there is a lot of variation at the ends of the alignment. Why do you think this is? (Hint: this is a question about genome assembly in disguise; take a look back at the contig that we produced in the assembly assignment if needed.)

Answer (3 points): I'm open to a number of possibilities here, but there is a simple answer. In the sequencing process, it is very rare to capture the substring falling exactly at the end of the genome, and more generally, the likelihood of getting nucleotides at the end of the genome is lower because there are fewer reads that can cover these positions. As a result, we may not be able to safely infer the nucleotides at the end of the sequence.

Now that we have explored the alignment data for November 3, let's check in on the virus a few weeks later. Follow the same steps to view the alignment for the December 8 file (`HCOV19-ENGLAND-081220.fasta`). You will see in this alignment that there are far more mutations in the sequenced genomes.

Exercise 5: Use the notion of a molecular clock discussed in class to explain why it is not surprising to see an increase in the number of viral mutations over time.

Answer (3 points): We know that mutations accumulate as organisms differ, and just because these mutations accumulate does not mean that they necessarily provide fitness to the organism involved. This was the "molecular clock" hypothesis of Zuckerkandl mentioned in class, with the example of hemoglobin being very similar in multiple organisms despite huge number of mutations.

What is worrisome about the increased number of mutations that you can see in the alignment is not that mutations are occurring, but that their frequency appears to be increasing *in the same column*. This is not what we would expect if mutations are simply occurring randomly, and it implies that the virus may be gaining mutations that are producing variants that are in some way "more fit". We will now zoom in on some of these mutations.

Profiling individual mutations

In this section, we will give an overview of three mutations that researchers have been monitoring in the gene that encodes the coronavirus's spike protein. We will focus in on this protein in the next challenge because the spike protein binds to the ACE2 enzyme on the surface of human cells; minor changes in bonding affinity can therefore have enormous consequences on the infectiousness of the virus.

N501-Y. (the ACE latcher)

This is a well-studied single-nucleotide mutation discovered in April 2020; it allows the coronavirus to more tightly fit the ACE2 receptor on human cells (the receptor that facilitates the virus's cell entry). Luckily, studies have shown the coronavirus vaccine is still efficacious against this mutation.[3]

The mutation occurs on the 501st amino acid of the spike protein, which corresponds to position 23063 of both the November 3 alignment and the December 8 alignment. (The annotation of the SARS-CoV-2 genome indicates that the spike protein starts at position 21563.) Note: If you look at the December 8 alignment, the mutation may not show up unless you zoom in.

Exercise 6: What was the presumed original nucleotide at this position? How many of the 100 November 3 genomes possessed the original nucleotide? What was the mutation, and how many of the 100 December 8 genomes have the mutation?

Answer (3 points; 1 for each part): The original nucleotide is adenine. On November 3, 98 of our sampled genomes had this nucleotide. The mutation changes an adenine to a thymine, and 44 of the 98 December 8 genomes had the mutation.

H69/H70-del. (the antibody butter)

This is a deletion present in many coronavirus lineages; it is therefore referred to as a *recurrent deletion region*. Although studies have confirmed this mutation *does* make the coronavirus more infectious[4], scientists are not entirely sure why. They speculate it may prevent antibodies from binding as tightly.

The mutation removes the 69th and 70th amino acid of the spike protein, which corresponds to positions 21765 through 21770 of both the November 3 and December 8 alignments.

Exercise 7: What is the nucleotide sequence that is deleted?

Answer (2 points): TACATG.

---

[3] https://www.medrxiv.org/content/10.1101/2021.01.19.21249592v1

[4] https://www.biorxiv.org/content/10.1101/2020.12.14.422555v4

Exercise 8: You know from class that just because we see gap symbols does not mean we can infer that a deletion occurred. It may be that these six nucleotides were *inserted* in an ancestral sequence. Use the November 3 alignment to argue why the mutation is most likely a deletion.

Answer (3 points): Only six of the November 3 genomes have the mutation, so it is most likely a deletion that is increasing in frequency over time.

P681-H. (the enzyme booster)

This is a mutation also present in many coronavirus lineages internationally. Scientists believe that this mutation makes it easier for human enzymes to prepare the spike protein for cell entry.

The mutation occurs on the 681st amino acid of the spike protein and at nucleotide position 23604 of our alignments.

Exercise 9: What was the original nucleotide at this position? How many of the November 3 genomes had it? What was the mutation, and how many of the December 8 genomes had picked up the mutation?

Answer (3 points; 1 for each part): The original nucleotide was cytosine; 99 of the November 3 genomes had it. The mutation changed the cytosine to an adenine, and 44 of the December 8 genomes accumulated it.

Comparing variant prevalence over time

What makes the above three mutations special is that they frequently occur *together* as part of a variant called **B.1.1.7** that you have probably heard about in the news.

From the exercises in the previous section, our hypothesis is that the prevalence of the B.1.1.7 variant is increasing over time. You will explore that hypothesis in this section by writing a computer program to determine the frequency of B.1.1.7 in each of our samples.

Note that just because the mutations occurred at the same positions in the November 3 and December 8 samples does not mean that they occurred in the same positions of all the samples. You should begin by looking at the November 10, November 27, and December 5 alignments to find the locations of the three mutations. (Remember that the November 17 data are corrupted.)

Exercise 10: For each of the five sampling dates, find the proportion of genomes in the sample having *all three* of the B.1.1.7 variants introduced in the previous section. Here is a series of steps you may like to follow.

1. Parse the file into a data structure of your choice (a common choice is a dictionary in which a key is a FASTA sequence header and the value is the sequence itself).
2. Initialize a counter.
3. For each genome in your data structure….
    a. Check all three mutation *locations*.
    b. If all locations feature the mutation *letter* you wrote down, increment the counter.
4. Output the counter divided by the total number of sequences in the sample.

Answer: 5 points, 1 point for each date.

Visualizing variant changes

You were likely wondering what the `HCOV19-ENGLAND-xxxx20-D.pim` files were. These are *pairwise percent identity* matrices; they are essentially inverse distance matrices, where $M_{i,j}$ represents how similar sequence *i* and sequence *j*: if $M_{i,j}$ is equal to 100%, then the sequences are identical, and if $M_{i,j}$ is equal to 0%, then the sequences have no commonality.

In the data folder, we have provided a Python script in the compressed file `heatmap.tgz`. This script generates **heat maps**[5] coloring the values in our `.pim` matrices according to how similar the corresponding values are. If you have Python installed, you simply need to unzip this file, run the script on all five relevant `.pim` files, and view the output images.

If you do not know how to run Python scripts locally, see the Appendix for instructions on running them on Andrew machines.

Exercise 11: Take a look at the output file of `HCOV19-ENGLAND-120820-D.pim`. What interesting/prominent features do you notice about this resulting heatmap? What could these features represent? How do they differ in the other heat maps? What can you conclude from these heat maps about the spread of SARS-CoV-2 in England?

Answer: 4 points, 1 point for a reasonable answer to each question.

[5] This section assumes that you are familiar with command line basics.

Appendix.

(running python scripts on the Andrew Machines)

Here, we will show you how to run the `heatmap.py` visualization script on the Andrew Machines, a set of remote computers hosted by Carnegie Mellon University and running RedHat Linux Enterprise (a text-based operating system). You can log into these remote machines from your computer using the command line / terminal.

If you've never accessed these machines before, no worries; there's nothing you need to do. All Carnegie Mellon University students automatically have an account.

Here are general instructions you need to follow. When typing in a command listed below, type each line separately and *without the dollar sign* (only the bolded text).

This shouldn't take more than 10-15 minutes.

1. First, download the compressed file `heatmap.tgz`.
    a. Place it on your `Desktop` or some easily accessible directory/folder.
    b. DO NOT unzip the file.
2. Open two separate command lines / terminal windows at the `Desktop` (or your desired directory).
    a. We will call these, `CMD1` and `CMD2`.
3. In `CMD1`, type the following command.

```
$ scp heatmap.tgz yourandrewid@unix.andrew.cmu.edu:~/private
```

4. In `CMD2`, log into your account on the Andrew Machines with *forwarding enabled* and navigate to the `private` folder. You can do this by typing the following commands.

```
$ ssh -Y yourandrewid@unix.andrew.cmu.edu
$ cd private
```

    a. Type the password that you use for Andrew Account.

5. Unzip the `heatmap.tgz` file and enter into the folder it created. You can do this by typing the following commands (all the following commands occur in `CMD2`, unless otherwise specified).

```
$ tar -xvf heatmap.tgz
$ cd b117-heatmap
```

6. Install the required modules for running the script. You may already have these modules installed from previous courses, but in the case you don't, type the following commands.

```
$ pip3 install --user numpy
$ pip3 install --user matplotlib
$ pip3 install --user pandas
$ pip3 install --user seaborn
```

7. Now we can run the script! Type the following command into the command line (replacing the ***xxxxxx*** with a date of your choice).

```
$  python3 heatmap.py HCOV19-ENGLAND-xxxxxx-D.pim
```

8. You should notice that a new image file has been created in the directory: `heatmap-xxxxxx.png` (you can see this by typing `ls` into the command line). This is the heatmap. To view the image on the command line, type the following command.

```
$  display heatmap-xxxxxx.png
```

   a. In the event this doesn't work (sometimes the Andrew Machines are finicky with image displaying), you can copy this file back to your computer and view it locally. You can do this by typing the following command into `CMD1` (don't forget the final period).

```
$    scp yourandrewid@unix.andrew.cmu.edu:~/private/b117-heatmap/heatmap-xxxxxx.png .
```

9. You are done! Repeat the instructions starting at **Step 7** if you want to generate more heatmaps for different days. Once you are completely finished with the task, type the following commands into `CMD2` to permanently delete everything.

```
$  cd ..
$  rm -r b117-heatmap/ heatmap.tgz ._b117-heatmap
$  exit
```