Computing for Medicine - Sequence Analysis

Dr. Jared Simpson
Department of Computer Science
University of Toronto
&
Ontario Institute for Cancer Research

1 Project

DNA sequencing is now applied to nearly all areas of biological research and medicine. Thousands of human genomes have been sequenced revealing patterns of genetic variation that improve our understanding of disease and how humans evolved and spread across the world. We've sequenced cancers arising from dozens of different tissues to learn about the mechanisms controlling cell division and how their dysfunction may lead to uncontrolled cell growth. In this project we will explore how DNA sequencing can be used to better understand the spread of a viral infection.

In late 2013 an outbreak of Ebola virus started in Guinea in West Africa. This outbreak was the largest Ebola outbreak to date and ultimately caused 11,000 deaths in Guinea, Sierra Leone and Liberia. A major international effort was made to monitor and control the spread of the virus and this included the use of DNA sequencing as an epidemiological tool. Ebola samples were taken from infected patients and sequenced, either in labs abroad or directly in the field using portable sequencing instruments. The genome of each sample was reconstructed from the sequencing data and compared to a reference sample from early in the outbreak to track mutations as they occurred and relate cases to each other.

In this project we will replicate this analysis using simplified simulated sequencing data. You will write code to reconstruct Ebola genomes from sequencing data, find mutations with respect to a reference strain, find cases that might be related and build a phylogenetic tree.

1.1 Input Data

The input data for this project (provided in the zip file with the starter code) is a set of 20 files, each containing DNA sequence reads from a single sample of Ebola. Each file of reads is in the FASTA format. The files are named according to where and when the sample was made.

1.2 Reconstructing a genome from sequencing reads

The first step of this project is to reconstruct the genome of each Ebola sample from the DNA sequencing data. When we sequence a genome, we fragment many copies of the genome into short pieces, then use a DNA sequencing instrument to read the sequence of each fragment. DNA sequencing instruments often make *errors* where it will mis-identify a nucleotide (for example the sequencer might mis-identify a `C` nucleotide as a `T`). To reconstruct a genome, we reverse this process - we take the sequence reads and try to compute the sequence of the genome. This task is computationally difficult and the subject of much research. To illustrate the general principles of this problem, we will work with simplified sequencing data for this project. We will assume that:

1. Each DNA sequencing read covers the entire genome
2. Only substitution errors occur

Figure 1 shows three sequencing reads (r_1, r_2, r_3) from a genome g. The sequencing errors are shown in red.

In this task we will write a program to reconstruct the sequence of the genome g from the reads $r_1, r_2, ..., r_m$. We'll do this by taking the *consensus* nucleotide at each position of the genome. The consensus nucleotide $g[i]$ is defined to be the nucleotide that occurs most often in the reads at

```
r1 CAGATAGTCGGATGTTATGAACCAGATATATA
r2 CAGACAGTCGGATGTTATGATCCAGATATGTA
r3 CAGATAGTCGGATGTTATAATCCAGATATATA
 ________________________________
 g CAGATAGTCGGATGTTATGATCCAGATATATA
```

Figure 1: An example of three sequencing reads from a small genome g.

postion i. For the example in figure 1 that consensus nucleotide $g[0]$ is `C` as all reads contain a `C` at position 0. The consensus nucleotide $g[4]$ is `T` as two reads have a T and only one read has a `C`.

The file `calculate_consensus.py` provides the framework for this exercise. The function `calculate_consensus_from_reads` will open one of the input files and parse the read sequences into a list of strings. You should write code to calculate the genome sequence from these reads. There is code provided to write the genome sequence you calculate to an output file. Once this code is completed, test it by running the function you wrote on one of the input reads file. When you are satisfied your code works, write code that will find all of the input reads files and calculate the genome sequence for each sample. You should end up with 20 output files, with the names `<sample_id>.genome.fasta`.

1.3 Finding mutations using a reference genome

Once we have calculated the genome sequence for each of the 20 Ebola samples, we want to analyze the genomes to see what they tell us about the outbreak. In this exercise we will compare the sequence of each genome to a *reference* genome. The reference genome (file `EM_079517.fasta`) is a sample that was sequenced early in the 2013 West African outbreak. By comparing the later samples to this early sample, we could calculate the rate at which the Ebola genome(s) are mutating, determine whether the proteins encoded in the genome have changed, and whether these protein changes contributed to the outbreak (e.g. by making the virus spread more rapidly).

In a file named `find_mutations.py`, write a program that loops over the genome sequences you computed in the first step and calculates the number of mutations that genome has with respect to the reference genome.

When comparing two positions between the genomes you should ignore bases that could not be sequenced (the nucleotide is letter 'N'). Write your output in the following format:

```
<sample_name> <position> <reference_base> <sample_base>
```

Here's an example output line for one genome:

```
EBOV_REDC483_MinION_GUI_Conakry_2015-07-12.genome.fasta 95 G A
```

Add to your program so that you can use it to answer the following questions:

1. What genome has the most mutations with respect to the reference?
2. What genome has the fewest mutations?
3. What mutation is most commonly seen? How many samples have this mutation?

Save your answers to these and the upcoming questions in a plaintext file named `seminar4.txt`. To ensure that the file is plaintext, use Wing 101 to create and edit the `.txt` file.

1.4 Clustering samples

Mutations tend to be rare; in any sample we only see around 20 nucleotides that are mutated in the Ebola genome which has almost 19,000 nucleotides in the sequence. As mutations are rare we might expect that any samples that have mutations in common might be related (e.g. they have a common ancestor sometime in the past). Analyzing the pattern of mutations shared between samples/cases can provide crucial insight into how the virus is spreading and direct control efforts. For example, when a new sample is sequenced we might want to search our database of samples to find the Ebola genome that is most similar (has the fewest number of differences) to the new sample.

Copy your `find_mutations.py` program that you wrote in 1.3 and modify it to create a related program `all_pairs.py`. First write a function that calculates the number of mutations between any two pairs of genomes, g_a and g_b. Next use this function to calculate the number of mutations between all pairs of samples. You need to answer the following questions:

1. What genome is closest to sample `EBOV_EM_COY_2015_017498_MinION_GUI_Forecariah_2015-06-08`?
2. How often do a pair of samples have the exact same sequence (no positions are different). What might this tell you about these cases?

You might be tempted to answer these questions by simply eyeballing the output of the 400 counts of all the pairs (including a genome with itself and duplicates.) Instead, use your developing programming skills to add to `all_pairs.py` so it computes the answers to the questions for you. You would need to do it this way if we had 2,000 samples instead of 20.

1.5 Building a phylogenetic tree

The patterns of shared mutations tells us something about the history of a set of samples. Samples that have high sequence similarity - that is, their sequences have few differences - likely had a common ancestral strain of the virus in the recent past. A way to organize and visualize these relationships is by building a *phylogenetic tree.*

In this part of the project we'll generate a phylogenetic tree for the Ebola genomes you constructed and use it to explore the relationship between samples. First make a single file containing all of the genome sequences by running the provided python script `merge_genomes.py`. This script will generate a file called `all_genomes.fasta`, containing all Ebola genomes that you calculated in step 1.2.

Next, open a web browser and go to https://www.ebi.ac.uk/Tools/phylogeny/clustalw2_phylogeny/. Select the "upload a file option" and select the `all_genomes.fasta` file you created just now. Select the "UPGMA" clustering method then hit submit and the website will begin to build a tree (it should only take around one minute).

Describe any patterns you see in the tree and what it might tell you about the outbreak.