Application

Parameters

k-mer size for contamination removal: As I decrease the k-mer size for contamination removal more reads were removed from the assembly due to contamination. I ended up choosing a value of k = 8 which had a somewhat stable percentage of reads removed. Further decreasing the value of k, understandably, led to many reads being removed, which did not seem likely in the context of the assembly parameters. In this case, k = 8 led to the removal of around 5.72% of all input reads.

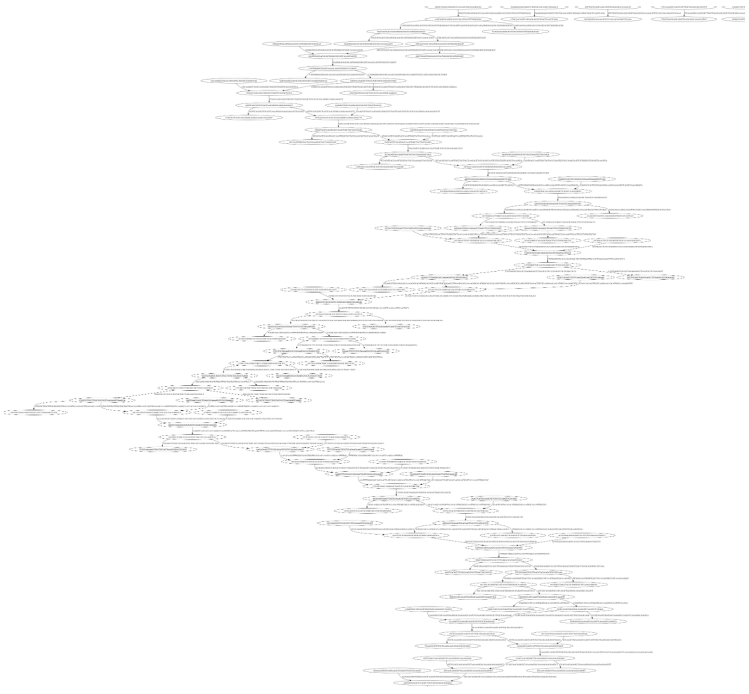
k-mer size for error correction: I used 15-mers for error correction. This value of k is large enough such that an erroneous read in our sequencing should only appear once in our reads. If we were to make k too large, it would be possible that a correct read only occurred once and we could improperly perform correction on that sequence.

t-value for correction: Because of our chosen value of k for correction, we can assume that any k-mer, or 15-mer, that occurs more than once is an erroneous read and thus required correction. Therefore, choosing a t-value of 2 makes the most sense.

d-value for correction: Further, for our 15-mers, we should be able to find the exact replacement for our inaccurate read due to our vast coverage of the genome. Thus, a value of d = 1 makes the most sense.

k-mer size for deBruijn graph assembly: I started out with higher values of k, such as 35, and incrementally decreased it to smaller values of k. Having a lower k would enable you to have fewer nodes in your graph - because there would be less possible k-1-mers. The reads for the COVID genome seemed to be of high coverage and values of k, such as 15, we were able to recover the genome completely. However, it should be noted that once the value of k became too low, multiple correct assemblies were generated as the graph was likely too simple to properly reconstruct the genome.

For fun, here is the dot graph that was produced by the deBruijn graph! It's very big, haha.



Waterman Statistics

What are the challenges to computing mean contig length from your assembler when the input reads have sequencing errors?

If we have erroneous reads, they will insert false values into the calculation of mean contig length. In a scenario where we had no read errors, we would see that the directed paths we observed in the deBruijn graph would each represent one contig. However, with erroneous reads, we can see that either incredibly short paths would be generated or multiple paths are generated over the same contig. That would confound the calculation of average contig length.

To better perform this calculation, we can try and filter out paths/contigs that seem as though they should not be counted. For example, we can remove extremely small contigs that would be unlikely as per the calculations we saw in class. Furthermore, we can try and remove contigs that are almost entirely identical. This process can be motivated by regions of the graph. For example, we can track the paths taken over a particular region and compare the nodes visited in the construction of the path. If there is almost a complete overlap, save for one difference at the beginning or end of a path, then we know that this contig is likely a repeat or a partial duplicate of the larger contig in that area.

Using maximum contig length as a "biased-up" proxy for mean contig length, how does mean contig length vary with coverage?

The length of the genome is around 9180 base pairs with a read length of 70. Then, for coverages of 1, 2, 3, and 4, we would need 185, 370, 555, and 740 reads respectively to achieve these coverages. These values were computed using the equation $c = nl/L$. Note that 185 is an approximation (the math yields 1 * 9181 / 50 = 183.62).

Coverage (Num Reads)	Biased Approximation for Mean Contig Length (k=25)
1 (185)	123
2 (370)	250
3 (555)	407
4 (740)	612

Understandably, as we increase our coverage, our mean contig length should increase. This follows the solution to Question 3 of Waterman Statistics.

How does mean contig length vary with k, the k-mer size you use to break down your reads (as described above)?

Let us fix our coverage at 10. From here, we can experiment with different values of k and see how this impacts our biased approximation for mean contig length.

k	Biased Approx. Mean Contig Length
12	7920
15	7604
20	4831
25	4312
30	1293

For a coverage of 10, as k increases to higher values, my mean contig length decreases. As we increase k, and maintain the same coverage, our graph will become more sparse and it is more likely that we are unable to create a path that traverses the entirety of the genome (i.e. the contig will be shorter).

If you have incorrect sequencing reads that are not corrected, this would have a greater impact on our sparse graphs for higher values of k. Then, I would likely trust the quality of the contigs that are created for lower values of k. It should be noted, though, that if values of k become too low, it might become impossible to actually reconstruct the graph itself.

With your answer to the second question in mind, evaluate the robustness of our answer to question 3 of Waterman Statistics. Where does it fail for the HIV genome? Is it ever a good approximation?

The equation from Question 3 states that the mean contig length will be $\frac{l}{a}(e^a - 1)$ for a coverage of a and a read length of l. Our assumption in this proof follows from the binomial distribution's convergence to a Poisson distribution. In particular, our lambda is the Poisson Distribution, a, is said to equal pn where p is the probability a read lies on a single base pair and n is the number of reads. This assumption is valid because we have an extremely small probability and a larger number of reads that balance out.

However, as we increase the coverage, our number of reads increases but our probability of a read aligning to a region of the genome will remain the same. Then, as we increase coverage, our assumption will break! We can even observe that for high coverage values that our exponential term will explode beyond the length of the actual genome.

However, if we maintain reasonable coverage and resolve that our probability is still somewhat small, it is possible that this equation still works for the HIV genome. Because the HIV genome is so small, it holds for very small coverage values (such as c = 3 where $\frac{50}{3}(e^3 - 1) = 318.09$ which is close to the observed value of 407.