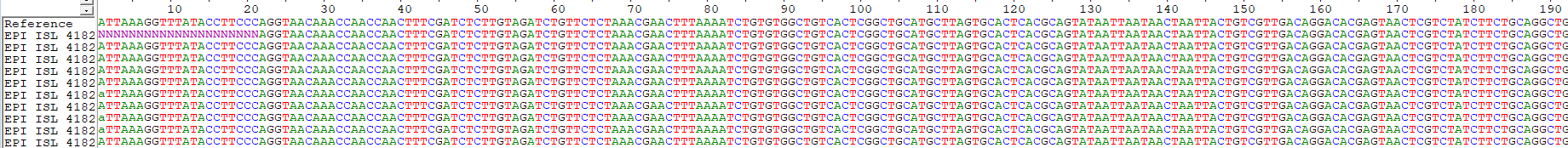
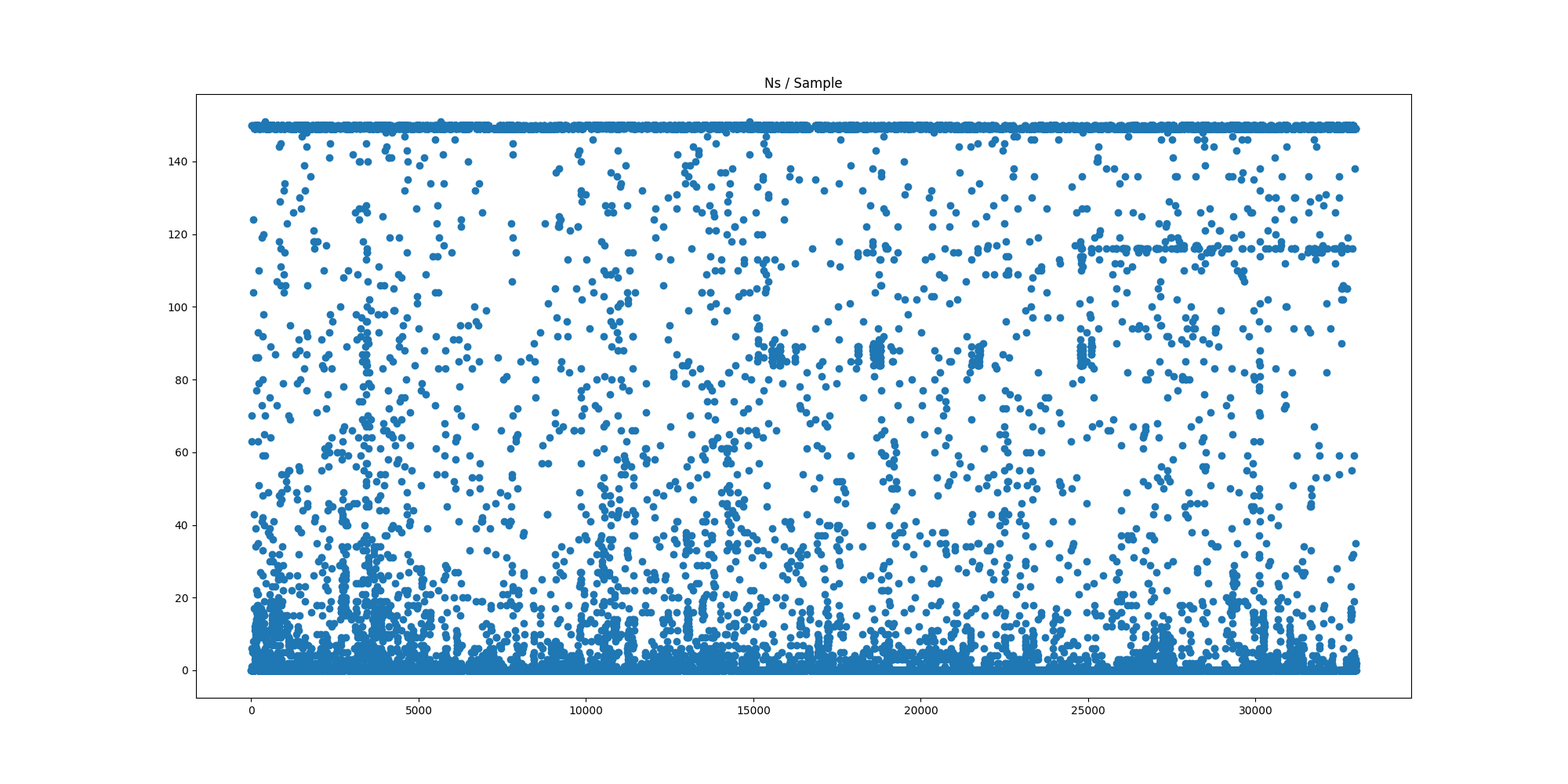
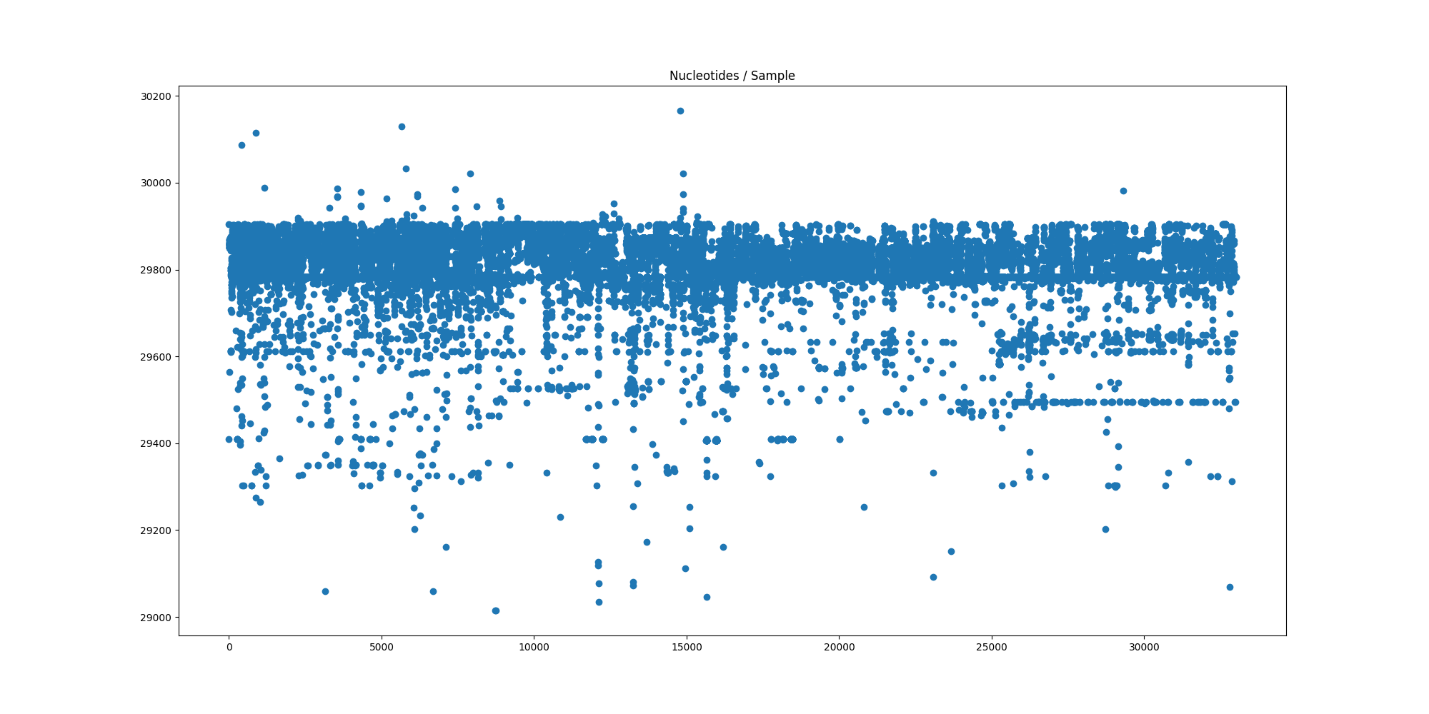
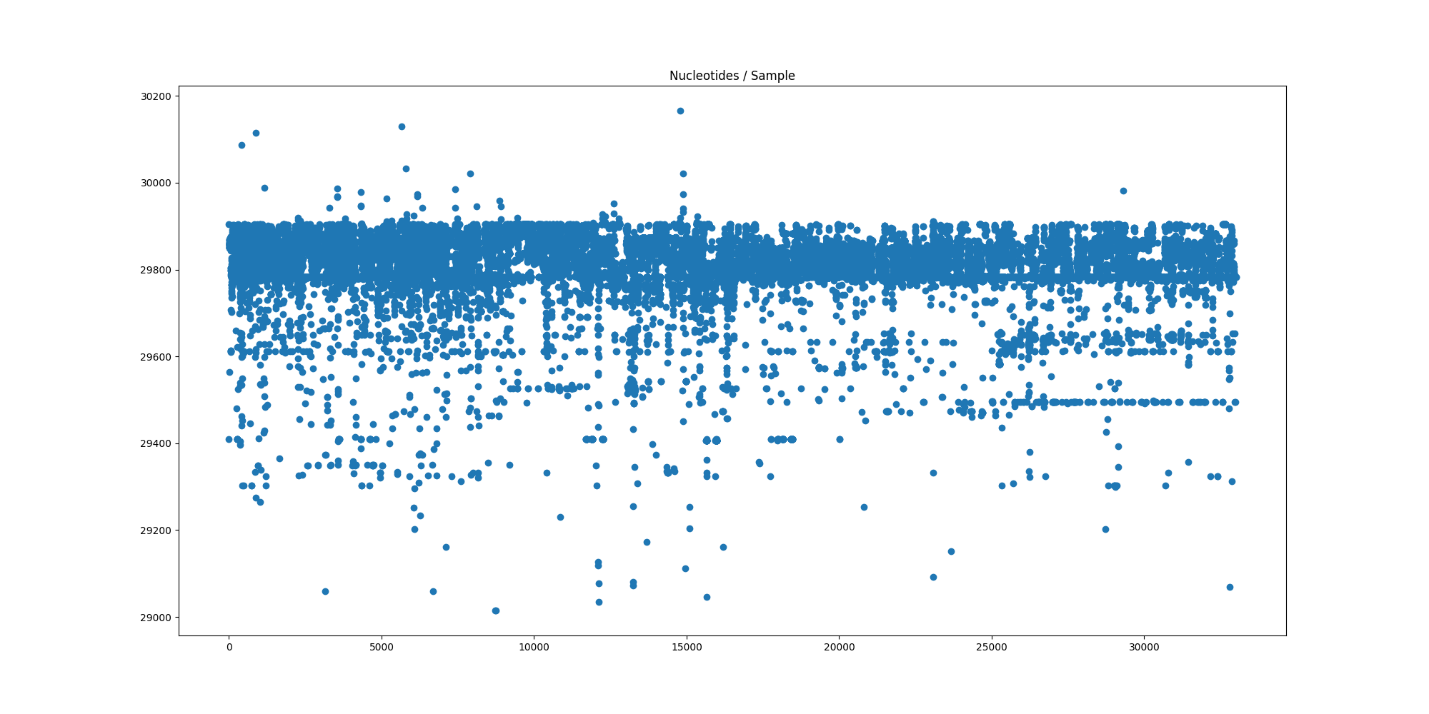
1. Goals and objectives
   1. Produce a fast and accurate sequence alignment
   2. Produce a fast and accurate pairwise alignment
   3. Build a tool to quickly characterize and sort good sequences from poor (N > 0.005% and Nucleotides < 29750) sequences.
2. Summary of results (~3 bullet points)
   1. The results for the sequence alignment were somewhat fast but the reference sequence would be moved to fit the rest of the sequences.
   2. Pairwise alignment was intended to fix the previous issue by comparing each sequence the reference sequence. This work except it was magnitudes slower.
   3. The tool was successful in sorting and removing poor sequences. Approximately 12,000 of the 33,000 were removed. This in can help reduce the time that it takes for a pairwise alignment.
3. Summary of methods  (2-5 bullet points)
   1. Libraries used are: Pandas, Bio, numpy, and matplotlib.pyplot
   2. Using bio.pairwise2 you can pairwise align two sequences relative to each other. These sequences can be exported to a file.
   3. The tool for separating samples takes all samples in and separates them as strings. Then by characters counts the corresponding “A”, ” T”, ”G”, ”C”, ”N”, and ”-“. If the value of N exceeds 0.005% of all nucleotides or A + T + G + C is less than 29750 the sequence is removed.
4. Key results (tables, figures, stats, lists, etc)

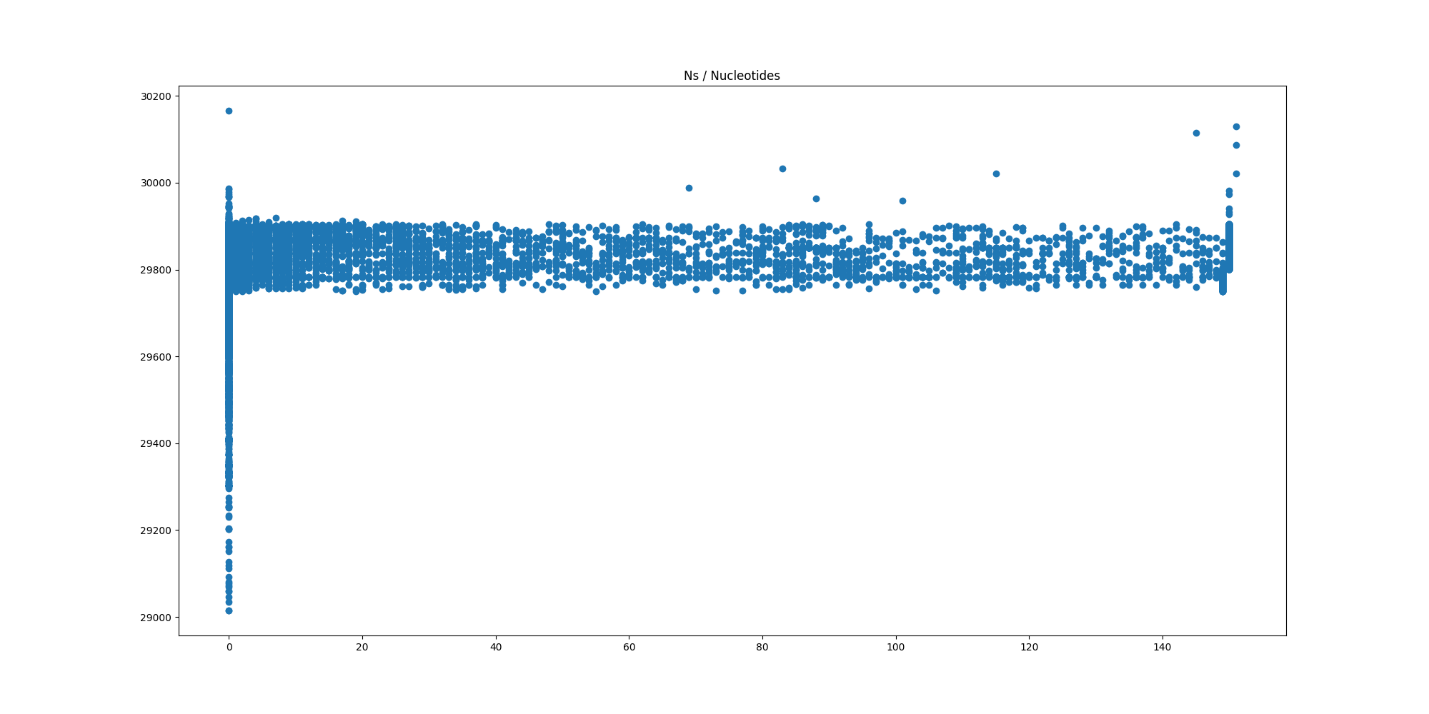


Sequence alignment for 10 features againts the reference genome.



This table shows samples over “N” values.



This table shows Samples over nucleotide lengths

This table shows Nucleotides over N count

1. Discussion  (2-5 bullet points)
   1. The month of September was good for experimenting with possible solutions for pairwise sequence alignment.
   2. It was very recently that we began sorting data by various metrics so I would predict that many more developments with this will be made in October.
2. Next steps (2-4 bullet points)
   1. Improve method for pairwise alignment
   2. Find more metrics for separating good and poor data.
   3. Looking at other metrics such as collection date and location for sorting data.