**Hackathon – shotgun metagenomics**

This hackathon problem deals with the following publication, which will be discussed briefly by Dr. Lozupone:

Hall, A.B., Yassour, M., Sauk, J. *et al.* A novel *Ruminococcus gnavus* clade enriched in inflammatory bowel disease patients. *Genome Med* **9,**103 (2017). <https://doi.org/10.1186/s13073-017-0490-5>

Among other findings, this paper describes how the presence of specific strains and genes in the gut microbiome can potentially discriminate between patients with Inflammatory Bowell Disease and Healthy control individuals. This is a longitudinal study, and some patients also have variance in the abundance of *Ruminococcus gnavus* strains over the course of sampling that appears to correlate with disease severity.

We are going to see if we can use homology based methods and kmer-based methods to generate data visualizations for exploring these associations. **You should work as a team or in teams. I would recommend teams of 2-3, but I am open to the entire class working as a team if the TAs agree that this can be productive**. All team members should be able to explain the choices made by the team during implementation, and the visualizations.

**Data** you will need to download ahead of time (can be manual download – this part does not need to be coded):

1. Supplementary table S1 from the paper (clinical data)
2. A table of SRA files associated with the project. One way to do this is using NCBI’s SRA Run Selector tool. Using the Bioproject accession for sequencing metagenomes reported in the paper, find the study of interest at the NCBI SRA database. Using the SRA’s Run Selector, generate an output table which includes at minimum the sequencing run identifiers (SRRxxxxxxx), the month of longitudinal sampling, and the patient identifiers (e.g. p8808).
3. *R gnavus* isolate database: <https://www.dropbox.com/sh/3dxl2vupk20ezui/AACmiJ5QkCR0Uh0fwoOtMbMIa/Ruminococcus_gnavus.tar.bz2>
   1. note that the database above comes with fasta files of contigs from the isolate genomes, as well as pre-build bowtie2 indices. You can use another tool for mapping if you like.

**Using these data, accomplish the following.** Write your code to take the following input:

* paths to the relevant input files from 1-3 above
* a single patient identifier (e.g. p8808)

Your code should achieve the following. Work with patient p8808 as an example, and compare to Fig. 2c for context.

1. Identify the SRA runs (SRRxxxxxx) that belongs to patient p8808.
2. Download reads from patient **p8808** as paired-end fastq files. (You might wish to use fasterq-dump, or use the ENA ftp site)
3. For each run:
   1. Potentially take a sample of reads to speed these steps.
   2. Using a tool of your choice, map reads from this run to the database of *R. gnavus* isolate genomes.
   3. Keep only the mapped reads.
4. Compare samples by creating a vector of read counts for each of the isolate genomes.
5. Compare samples by creating a vector of kmer frequencies. The choice of k is up to you.
   1. Do this for all reads (or a sample of reads)
   2. Do this only for reads mapping to any *R. gnavis* genome.
6. Cluster the patient samples by the data generated in 4., then try with the data from 5a., and finally try with the data from 5b. The choice of clustering method or dimension reduction technique is up to you.
7. Produce a visualization of the metadata (Supplementary Table S1) overlaid on your clustering, to argue for or against the ability to detect clinically relevant changes in microbial community via each of the approaches in 4., 5a., or 5b.

**Some additional things to consider:**

Data volume will be large, and run times for mappings might also be on the long side. Get your code working with a subset of reads and a subset of samples first.

Start with simple methods first. Even if you’d like to try more complex clustering, write your code in a modular way that allows for the clustering method to be dropped in and out easily. Start with a simple clustering method to implement and visualize and get that working before moving to more complicated approaches.

Parameterize your k-mer size, and choose a manageable k-mer size to begin with before exploring the benefit of other k-mer sizes.

It’s okay to use existing libraries/modules to implement clustering and visualizations.

The language you choose to use for implementation is up to you. Not everyone will be equally strong in any given language (including your TAs and instructors!). Carefully commented code is important for communication with your teammates and with the instructional team.