180130 - Updated workflow:

Tuesday, January 30, 2018 1:15 PM

1. Run "download_gut.sh < input_file.txt". (This script needs to be replaced with a better Python one). It takes as input a five-column tab-separated file and:
   a. Ignores the header
   b. Downloads GBFF, GFF, FNA, and feature_table for each genome in the input file
   c. Unzips the files
   d. Makes bioproject.txt (containing the bioproject #), organism.txt (containing the organism name separated by underscores which is also the directory name)
   e. Makes an all_organisms.txt file containing each organism (separated by underscores, which is also the directory name), one per line.

2. Run convert_and_generate_contigs_db.sh. This contains the steps to:
   a. Write an external_genomes file (used for building a genomes storage prior to pangenomic analysis)
   b. Run gff_to_external_calls.py
   c. Run smiplify_fasta.py
   d. Generate a contigs database
   e. Run HMM to identify single-copy genes
   f. Imports locus tags into contigs database

3. Run command to generate NCBI COGs for each contigs database. (HAS INSTALL ISSUES, DESCRIBE)
   for i in `cat all_organisms.txt`; do
   anvi-run-ncbi-cogs -c ${i}/CONTIGS.db --cog-data-dir
   /home/bmerrill/user_data/Auxiliary_Data/anvio-data/COG -T 16 --sensitive;
   done

4. Prep files for GhostKOALA, run GhostKOALA, and separate output file.
   a. Generate protein FASTA files for each genome.
      for i in `cat all_organisms.txt`; do anvi-get-aa-sequences-for-gene-calls -c ${i}/CONTIGS.db -o ${i}/${i}-protein-sequences.fa; sed -i "s/>/>${i}_/g" ${i}/${i}-protein-sequences.fa; done
   b. Concatenate all outputs into one file. Max file size is 300 MB.
      for i in `cat all_organisms.txt`; do cat ${i}/${i}-protein-sequences.fa >> all_protein_sequences.fa; done
   c. Upload all_protein_sequences.fa to http://www.kegg.jp/ghostkoala/ and submit the job.
   d. When done, download the file "user_ko.txt".
   e. Separate user_ko.txt into individual files for each genome. BE CAREFUL THAT NO user_ko.txt FILE EXISTS OR IT'LL APPEND TO IT!
      for i in `cat all_organisms.txt`; do grep ${i} all_ghost_koala_results.txt | sed "s/ ${i}/genecall/g" >> ${i}/user_ko.txt; done

5. Run InterProScan. (HAD SOME INSTALL ISSUES, DESCRIBE). This takes a minimum of 30 mins per bacterial genome.
   a. With python 2.7 in the PATH, run:
      for i in `cat all_organisms.txt`; do
      /home/bmerrill/user_data/Auxiliary_Data/interproscan/interproscan-5.27-66.0/interproscan.sh -cpu 16 -f tsv --goterms --iprlookup --pathways -i ${i}/${i}-protein-sequences.fa -o ${i}/interproscan-results.txt; done

6. Combine GhostKOALA and InterProScan annotations, import into Anvi'o. (GHOSTKOALAPARSER

MAY HAVE SPECIFIC INSTALLATION REQUIREMENTS)

a. Run the following command (with ghostkoala environment activated):
   for i in `cat all_organisms.txt`; do
   KEGG-to-anvio --KeggDB
   /home/bmerrill/Applications/GhostKoalaParser/samples/KO_Orthology_ko00001.txt -i
   ${i}/user_ko.txt --interproscan ${i}/interproscan-results.txt -o ${i}/KeggAnnotations-
   AnviImportable.txt; done

b. Remove duplicate entries from KeggAnnotations-AnviImportable.txt. Run:
   for i in `cat all_organisms.txt`; do head -1 ${i}/KeggAnnotations-AnviImportable.txt >
   ${i}/KeggAnnotations-AnviImportable_unique.txt; tail -n+2 ${i}/KeggAnnotations-
   AnviImportable.txt | sort -u -t't' >> ${i}/KeggAnnotations-AnviImportable_unique.txt; done

c. Run the following command (with the anvio3 environment activated):
   for i in `cat all_organisms.txt`; do anvi-import-functions -c ${i}/CONTIGS.db -i
   ${i}/KeggAnnotations-AnviImportable_unique.txt; done

7. Generate Anvi'o genomes storage database. The file "anvio_external_genomes.txt" contains each genome name and the path to its contigs database. For this step, we may need to look into whether or not to include "partial" genes. These are genes that weren't translated because they were pseudogenes, were hanging off the edge of a contig or scaffold, etc. Right now, we're including them.

   anvi-gen-genomes-storage -e anvio_external_genomes.txt -o BACTEROIDES-GENOMES.h5 --gene-caller "NCBI Genbank"

8. Run anvi-pan-genome. PARAMETERS HERE NEED TO BE TESTED.
   time anvi-pan-genome -g BACTEROIDES-GENOMES.h5 -n Bacteroides_12_genomes_test2 -T 16 -o Bacteroides_12_genomes_test1 --sensitive

9. Export a summary file from the pan-genome analysis. This matrix will be analyzed in comparison to metabolites. See http://merenlab.org/tutorials/infant-gut/#a-pangenomic-analysis.

   The higher the number for --mcl-inflation is the (more) clusters are generated. The default is 2.

   I have a database and a table you can play with. You can find the database on the lab server at "/LAB_DATA/SHARE/micro_metabolites/bmerrill/gut_only/Bacteroides_12_genomes_blastp". To visualize, activate your anvio environment and change to this directory. Then run "anvi-display-pan -p Bacteroides_12_genomes_blastp-PAN.db -s Bacteroides_12_genomes_blastp-SAMPLES.db -g ../BACTEROIDES-GENOMES.h5".

   From <https://outlook.office.com/owa/?realm=stanford.edu&path=/mail/search>

NOTES:
For the 12 Bacteroides genomes, with --mcl-inflation set at 2, I got ~10908 protein clusters. With --mcl-inflation set at 6, I got 11,195 protein clusters (took 20 mins but reran on existing directory). With mcl-inflation set at 2 and minbit at 0.3, I got 8963 protein clusters (took 11 mins)

Minbit:

The "minbit" criteria emphasizes strong hits over the entirety of the smaller protein (so it can pick up pseudogenes but is less sensitive to events such as gene fusions);
From <https://github.com/mattb112885/clusterDbAnalysis/wiki/Building-your-database-2---mcl-clustering>

Using BLASTP doesn't work if incomplete proteins are used, I think…. So I add the flag "--exclude-partial-gene-calls" and run as follows:
time anvi-pan-genome -g BACTEROIDES-GENOMES.h5 -n Bacteroides_12_genomes_blastp -T 14 -o Bacteroides_12_genomes_blastp --use-ncbi-blast --exclude-partial-gene-calls

Generated 10952 protein clusters, took 40 mins.

Generating a summary file:
There will DEFINITELY be some work to do to get Will's magic matrix generated…
Maybe I can start by exporting the view data table??? That seems better. And then overlay concatenated functions on top or something? Who knows.

An important thing to investigate:
How does having two sets of gene calls around affect… anything? HMMs? Calling functions?!?
Investigate this, because it may be that, even though I'm downloading from RefSeq, that there are some missed gene calls. It's possible that we'd get a more consistent genes to m/z correlation if the same gene caller was done on all genomes. Worth a shot sometime!

To do:
Run mash clustering through dRep:
dRep compare_wf -p 16 -ms 100000 --SkipSecondary analyses/dRep_mash_100k -g finished_genomes/*/*.fna

At a threshold of 0.9, 203 genomes cluster down to 141 clusters.

dRep compare_wf -p 16 -ms 100000 --SkipSecondary -pa 0.85 analyses/dRep_mash_100k_85_cluster/ -g finished_genomes/*/*.fna

At a threshold of 0.85, 203 genomes cluster down to 129 clusters.

At a threshold of 0.85, 203 genomes cluster down to 105 clusters.

Next, I need to find out how different gene calls for the same organism behaves downstream…

For assembled organisms:
anvi-script-reformat-fasta -l 2000 -o Anaerococcus_lactolyticus_CC31C_scaffolds.fa scaffolds.fasta
sed -i -e 's/[^a-zA-Z0-9>]/_/g' Anaerococcus_lactolyticus_CC31C_scaffolds.fa

Make contigs database