# Assignment 8: Metagenomics

**Introduction**
Assembly of genomic fragments and gene annotation are important steps in any large-scale metagenomic study. One technique in particular, functional metagenomic selections [1], results in metagenomic fragments after assembly [2] that are often 1-5kb in length and are enriched for genes that encode for particular functions. You have performed functional metagenomic selections to enrich for antibiotic resistance functions found in bacteria residing in the human gut environment [3]. Your goal is to identify open reading frames (ORFs) and annotate antibiotic resistance functions. However, you realize that there are many different methods for accomplishing this goal and want to benchmark them against each other to determine the optimal pipeline for your analysis.

The assembled contigs can be found in `/home/assignments/assignment8/contigs.fna`.

**Part 1.**
Write a script called `call_orfs.py` that takes a fasta of contigs as input, scans all six reading frames for ORFs that start with an ATG, end with a STOP codon, and are >100bp in length (not including the stop codon), and **reports the longest ORF**. Since a stop codon does not code for a protein, it shouldn't count towards the length of an ORF. The script should create two output files:

- `all_orfs.fna`: a fasta formatted file with nucleotide sequences from predicted ORFs
- `all_proteins.faa`: a fasta formatted file with amino acid sequences from predicted ORFs that have been translated into an amino acid sequence

NOTE: A sequence record in a **FASTA format**—as you know by now—consists of a **single-line sequence description** (sequence name), **followed by line(s) of sequence data**. The first character of the sequence description line is a greater-than (">") symbol. Please use your prudence to define an appropriate sequence description, and **highlight** it in your code comments.

The usage of `call_orfs.py` will be:

```
$ python3 call_orfs.py <contigs file>
```

**Question 1:**
How many ORFs were predicted using the described method?

**Part 2.**
You realize that there could be more sophisticated algorithms out there for predicting ORFs in metagenomic DNA that take into account different start codon usage patterns across bacteria. You therefore decide to not reinvent the wheel and use an already developed gene caller for metagenomic DNA called MetaGeneMark. Read the instructions for running the software in `/home/assignments/assignment8/MetaGeneMark/README.txt`. We have provided version 2.8 of MetaGeneMark for you to use to identify open reading frames.

Run MetaGeneMark to predict ORFs and output them in General Feature Format (GFF) format [4]. To do this, run:

```
 $ gmhmmp -a -d -f G -m
/home/assignments/assignment8/MetaGeneMark/MetaGeneMark_v1.mod -o
mgm_predictions.gff /home/assignments/assignment8/contigs.fna
```

An explanation of each flag can be found by running `gmhmmp` without any input. This command should produce:

- `mgm_predictions.gff`: a GFF formatted file with gene predictions from MetaGeneMark

Finally, we want to parse out the nucleotide and protein sequences from the ORF predictions generated by MetaGeneMark, just as we did previously. MetaGeneMark provides two helper Perl scripts to do this:

- `/home/assignments/assignment8/MetaGeneMark/nt_from_gff.pl`
- `/home/assignments/assignment8/MetaGeneMark/aa_from_gff.pl`

Read MetaGeneMark's README (`/home/assignments/assignment8/MetaGeneMark/README`) to determine how to take the predicted ORFs and produce the following files:

- `mgm_orfs.fna`: a fasta formatted file with nucleotide sequences from predicted ORFs
- `mgm_orfs.faa`: a fasta formatted file with amino acid sequences from predicted ORFs

OPTIONAL: Write your own script `gff_to_nt_aa.py` that takes in the `mgm_predictions.gff` predictions file and writes the nucleotide and amino acid sequences from the predicted ORFs in the form of two FASTA formated files—.fna and .faa respectively.

**Question 2**:
Explain what each of the flags for the script `gmhmmp` are doing. Would you have added or dropped any flags for your particular problem or use?

**Question 3**:
How many ORFs were predicted using MetaGeneMark?

**Part 3.**
Write a script, `compare_orf_callers.py`, that compares the predicted ORFs from your `call_orfs.py` script to those called by MetaGeneMark. The script should output the ORFs found in both the output of `call_orfs.py` and MetaGeneMark, the ORFs unique to the output of `call_orfs.py`, the ORFs unique to MetaGeneMark, and the number of ORFs in each of these three categories.

The usage of `compare_orf_callers.py` will be:

```
$ python3 compare_orf_callers.py <fasta 1> <fasta 2>
```

**Question 4**:
How many ORFs were identified by both MetaGeneMark and your custom ORF caller? How many ORFs were unique to your custom ORF caller? How many ORFs were unique to MetaGeneMark? Please interpret your results and explain why the intersection and set differences are large/small.

**Part 4.**
Now that you have identified ORFs in the assembled contigs, you want to be able to assign function to these genes. The most common way to identify antibiotic resistance genes is to use pairwise sequence alignment through BLAST. Since we specifically selected for antibiotic resistance functions, we will annotate the amino acid sequences predicted by MetaGeneMark in Part 2 by BLASTing the sequences to the Comprehensive Antibiotic Resistance Database (CARD) [5]. The CARD database is located here: `/db/CARD.faa`. To run `blastp`, enter:

```
$ blastp -db /db/CARD.faa -query mgm_orfs.faa -out blast_to_card.txt
-outfmt "6 qseqid sseqid pident length mismatch gapopen qstart qend sstart
send evalue bitscore slen stitle"
```

This command will output:

- `blast_to_card.txt`: BLAST output in outfmt 6 with predicted genes against the CARD database

Now write a script, `count_ar_genes_from_blast.py`, that takes `blast_to_card.txt` as input and calculates the number of antibiotic resistance genes identified in your contigs using an identity threshold of >80% amino acid identity over >85% of the subject sequence.

The usage of `count_ar_genes_from_blast.py` will be:

```
$ python3 count_ar_genes_from_blast.py <blast output>
```

**Question 5**:
Explain briefly (in a few words) what each one of the parameters mean in:

```
$ blastp -db /db/CARD.faa -query mgm_orfs.faa -out blast_to_card.txt
-outfmt "6 qseqid sseqid pident length mismatch gapopen qstart qend sstart
send evalue bitscore slen stitle"
```

**Question 6**:
How many unique predicted antibiotic resistance genes were identified using BLAST against the CARD database? How many survive the filtering in your Python script?

**Part 5**.
Often, functional metagenomic selections identify antibiotic resistance genes that have low identity when compared to any known genes [3]. These genes are not identified using pairwise sequence alignment. Profile hidden Markov models (profile HMMs) have been shown to improve annotation of remotely homologous genes [6] and have recently been applied to genes with antibiotic resistance functions [7]. Use the software HMMER [8] to align your predicted proteins to the database of profile HMMs with this command:

```
$ hmmscan --cut_ga --tblout resfams_annotations.txt
/home/assignments/assignment8/Resfams/resfams.hmm mgm_orfs.faa
```

This will output:

- `resfams_annotations.txt`: an output file from `hmmscan` containing annotation predictions by Resfams

Write a script, `count_ar_genes_from_resfams.py`, that takes `resfams_annotations.txt` as input and calculates the number of antibiotic resistance genes identified in your contigs.

The usage of `count_ar_genes_from_resfams.py` will be:

```
$ python3 count_ar_genes_from_resfams.py <hmmscan output>
```

**Question 7:**
Mention two uses of HMMER, and how you would go about executing it. You can access the documentation here: HMMER userguide.

**Question 8:**
How many antibiotic resistance genes were annotated by Resfams? How does this number compare with the number of antibiotic resistance genes identified using BLAST (question 5)?

**What to turn in**

- A `README.txt` with the answers to the questions and the commands you used to answer the questions.
- A commented `call_orfs.py` to identify ORFs in a fasta of contigs
- A commented `compare_orf_callers.py` to compare MetaGeneMark output to your `call_orfs.py` output.
- A commented `count_ar_genes_from_blast.py` that filters your blast output.
- A commented `count_ar_genes_from_resfams.py` that counts the number of genes in a hmmscan output file.
- Optional: a commented `gff_to_nt_aa.py` that takes a MetaGeneMark GFF file and outputs the nucleotide and amino acid sequences from the predicted ORFs.
- All files created from the above scripts or commands: `all_orfs.fna`, `all_proteins.faa`, `mgm_predictions.gff`, `mgm_orfs.fna`, `mgm_orfs.faa`, `mgm_predictions.gff`, `blast_to_card.txt`, and `resfams_annotations.txt`

REFERENCES

[1] Novel Resistance Functions Uncovered Using Functional Metagenomic Investigations of Resistance Reservoirs. Pehrsson EC, Forsberg KJ, Gibson MK, Ahmadi S and Dantas G. Front Microbiol. 2013; 4: 145
[2] The Shared Antibiotic Resistome of Soil Bacteria and Human Pathogens. Forsberg KJ*, Reyes A*, Wang B, Selleck EM, Sommer MOA, Dantas G. Science. 2012; 337 (6097): 1107
[3] Functional Characterization of the Antibiotic Resistance Reservoir in the Human Microflora. Sommer MOA*, Dantas G*, Church GM. Science. 2009; 325 (5944): 1128
[4] https://www.sanger.ac.uk/resources/software/gff/spec.html
[5] McArthur et al. 2013. The Comprehensive Antibiotic Resistance Database. Antimicrobial Agents and Chemotherapy, 57, 3348-3357.
[7] Profile Hidden Markov Models. S. R. Eddy. Bioinformatics, 14:755-763, 1998.
[8] http://hmmer.org/