# Practical: Genomic Analysis (Morning)

## Overview

*Step 1: Verify species with KmerFinder*

*Step 2: Detect AMR genes and mutation with ResFinder*

*Step 3: Identify replicons with PlasmidFinder*

## How to use this document

Conda environment with required software: **cgecore 1.5.5, tabulate 0.7.7, BLAST, KMA, resfinder 4.3.1**

Software commands are highlighted in grey. Arguments must be filled in manually (so don’t copy-paste the commands!).

In this practical, you will predict species and identify AMR and replicons in sequence data.

* Make directories to provide structure
* Name your directories and files in a manner that clearly indicates the content
* Keep an eye on where your output-files are going.
* Use the list command (ls) excessively.

### Step 1: Verify species

Verification of species is a useful secondary quality check. Even if a sequenced isolate is within expected parameters and without any contamination it may have been misidentified in the lab.

* What does the result look like?
* Is the prediction unambiguous and as expected?
* Why do we see hits for other species in our results?
* What can we do to make sure the species prediction is correct?

#check help-info for kmerfinder. Why do we need both a database and a taxonomy file?

python3 path/to/program/kmerfinder.py -h

#run the 1st sample

python3 path/to/program/kmerfinder.py -i [input files] -o [output folder]

### Step 2: Detect AMR genes and mutations

Genes conferring resistance to antimicrobials in the strains is valuable information both to guide treatment in infection and for surveillance of potential risks to public health. Annotating these in the strains is crucial information for combating an outbreak.

* What information does the tool produce?
* What is the difference between using fastq and fasta files?
* Why do we treat mutations differently from genes?
* Do we see any indication that the isolates are related from their AMR profile?
* What is the species information required for?

#check help-info for ResFinder, what are the default identity and coverage thresholds?

python3 path/to/program/resfinder.py -h

#run the 1st sample, what does flags -acq and -c do?

python3 -m resfinder.py -ifq/ifa [input files] -o [output folder] -s [species] -acq -c

#check the output files

less [path to output fodler]/pheno\_table\_[species].txt

less [path to output fodler]/Resfinder\_results.txt

less [path to output fodler]/Pointfinder\_results.txt

### Step 2: Detect AMR genes and mutations

Genes conferring resistance to antimicrobials in the strains is valuable information both to guide treatment in infection and for surveillance of potential risks to public health. Annotating these in the strains is crucial information for combating an outbreak.

* What information does the tool produce?
* What is the difference between using fastq and fasta files?
* Why do we treat mutations differently from genes?
* Do we see any indication that the isolates are related from their AMR profile?
* What is the species information required for?

#check help-info for ResFinder, what are the default identity and coverage thresholds?

python3 path/to/program/resfinder.py -h

#run the 1st sample, what does flags -acq and -c do?

python3 -m resfinder.py -ifq/ifa [input files] -o [output folder] -s [species] -acq -c

#check the output files

less [path to output fodler]/pheno\_table\_[species].txt

less [path to output fodler]/Resfinder\_results.txt

less [path to output fodler]/Pointfinder\_results.txt

### Step 3: Plasmids

Mobile genetic elements (such as plasmids) move around cells potentially affecting the severity of an outbreak or infection.

* What information is produced?
* What is the difference between using fastq and fasta files?
* What thresholds should we use and why?
* Does these findings carry any significance with our AMR predictions?

#check help-info for plasmidfinder, what databases are available? what does the -x option do?

python3 path/to/program/plasmidfinder.py -h

#run the 1st sample, what does flags -acq and -c do?

python3 plasmidfinder.py -i [input files] -o [output folder]

#check the output files