# PRACTICAL: Taxonomic profiling

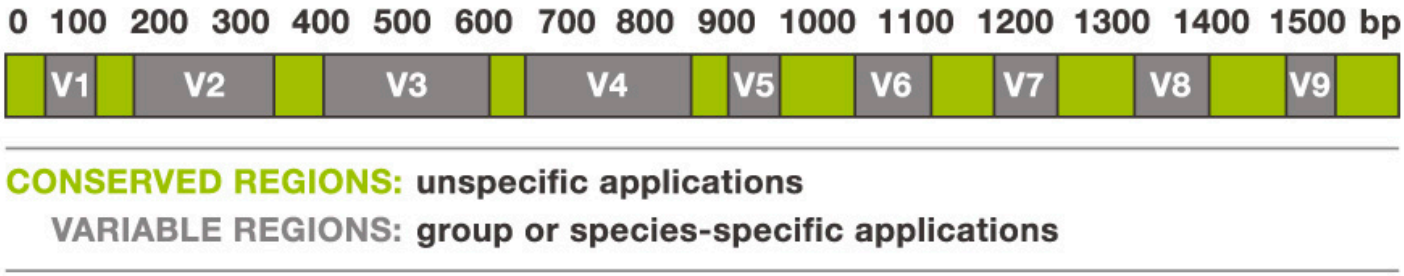
In this assignment we will continue to work with the same whole genome shotgun dataset from the previous exercise. When doing a taxonomic classification, we want to find out what kind of organisms the sample contains. Who is there? This assignment will take you through a workflow from raw reads to a classification of organisms present in the dataset.

The aim of this assignment is to get familiarized with the essential steps required to create a taxonomic classification. For simplicity, we will use the rawdata from yesterday in this exercise (not the trimmed and filtered data).

### This exercise consists of four parts:

1. Taxonomic classification using reads with 16S rRNA
2. Taxonomic classification using a reference database of protein sequences
3. Taxonomic classification using a K-mer based method
4. Taxonomic classification of contigs

## 1. Taxonomic classification using reads with 16S rRNA




💡 To do a taxonomic classification based on 16S rRNA sequences we need to get hold of all the reads coding for rRNA in the sample. Since these sequences are relatively conserved and present in all prokaryotic species, they can answer the question “Who is there?” for our sample.

I] Move to the directory `/practical/3` where you will do this part of the exercise and view what is in the directory. It should be empty.

```
cd /practical/3
```

II] We are going to use the same rawdata as yesterday. Here is a tip to save space: instead of copiying the files, make a symbolic link to the files with the command `ln -s path/to/directory/files .`. `.` means 'here'.

```
ln -s ../1/rawdata/sample_R1.fastq.gz .
ln -s ../1/rawdata/sample_R2.fastq.gz .
```

First we need to predict/find the ribosomal RNA sequences. The script **predict_16s.py** takes a single FASTA file as input. The script accepts both FASTA and (uncompressed) FASTQ as input. In this exercise, we want you use FASTA as input! You should also concatenate the forward (R1) and the reverse (R2) reads before predicting the rRNA sequences. There are several way to do this. Yesterday you both converted files from FASTQ to FASTA and you concatenated files.

III] You can for example use `seqkit fq2fa` to do this before merging:

```
seqkit fq2fa sample_R1.fastq.gz -o sample_R1.fasta

seqkit fq2fa sample_R2.fastq.gz -o sample_R2.fasta

cat *.fasta > sample_catAll.fasta
```

OR:

```
zcat sample_R1.fastq.gz sample_R2.fastq.gz | seqkit fq2fa -o sample_catAll.fasta
```

The 16S rRNA gene is ubiquitous among bacteria and is the most used target gene in metagenomic amplicon sequencing. We want to identify all sequence reads that contain parts of the 16S rRNA gene. The script **filter_16s_extended.py** uses **HMMER** with a 16S rRNA profile that very fast identify reads that contain parts of the 16S rRNA gene. You can find the script here.

IV] Run the script **filter_16s_extended.py** on the concatenated FASTA file where you only keep sequence reads with bacterial 16S rRNA. Remove those part of the reads that does not belong to the 16S rRNA gene. Name the output file `sample_16S.fasta` :

```
filter_16s_extended.py sample_catAll.fasta -b -n 2 -o sample_16S.fasta
```

❓ What does the -b flag do?

▶ **Solution** - Click to expand

❓ How many reads consists of 16S rRNA sequences?

▶ **Solution** - Click to expand

---

# MAPseq v1.1 (April 2017)

*improving speed, accuracy and consistency in metagenomic ribosomal RNA analysis*

💡 **MAPseq** is a set of fast and accurate sequence read classification tools designed to assign taxonomy and OTU classifications to ribosomal RNA sequences. This is done by using a reference set of full-length ribosomal RNA sequences for which known taxonomies are known, and for which a set of high quality OTU clusters has been previously generated. For each read, the best guess and correspoding confidence in the assignment is shown at each taxonomic and OTU level. The tool is available here.

VI] Run **MAPseq** on the FASTA file containing reads with 16S rRNA sequences.

```
mapseq sample_16S.fasta > sample_16S_mapseq.txt
```

💡 The default output format is a large table. Each line indicates a taxonomic classification of the read and confidence values for each of the taxonomic levels are written to the columns. You can use **Krona** to visualize the results. **Krona** allows hierarchical data to be explored with zooming, multi-layered pie charts. **Krona** charts can be created using an **Excel** template or **KronaTools**, which includes support for several bioinformatics tools and raw data formats. The interactive charts are self-contained and can be viewed with any modern web browser.


In order to visualize the results in **Krona**, you first need to convert the **MAPseq** results into a file ready for import in **Krona Tools**. Scripts to make the format conversion are found here.

VII] Make the 16S rRNA mappings ready for **Krona** import:

```
python ~/Programs/scripts/mapseq2krona.py -f sample_16S_mapseq.txt > sample_16StoKr
```

VIII] **Krona Tools** is one of many ways to visualize a classification. It is a set of perl scripts that can generate a pie chart of many different inputs that can be browsed interactively. Read more about **Krona Tools** here. Convert the file `sample\_16StoKrona.txt` to a HTML file using the **ImportText.pl** script in **Krona Tools**:

```
ktImportText -o sample_16S.html sample_16StoKrona.txt
```

IX] Open the Krona chart in your web browser by clicking on it in the **File browser** (left panel).

❓ What is the most abundant genus and is the relative abundance of this genus?

▶ **Solution** - Click to expand

❓ Yesterday you saw that the specie with the largest number of contigs align to the the genome was *Aliivibrio logei*. Why do you think you don't find it in this analysis?

▶ **Solution** - Click to expand

**Progress tracker**

Part 1 finished

# 2. Taxonomic classification using a reference database of protein sequences


💡 **Kaiju** is a program for the taxonomic classification of high-throughput sequencing reads, e.g., Illumina or Roche/454, from whole-genome sequencing of metagenomic DNA. Reads are directly assigned to taxa using the NCBI taxonomy and a reference database of protein sequences from microbial and viral genomes. You can read more about the tool here.

As true for most taxonomic classification tools, your data is screened against a database of known sequences in order to classify your data. Before you can perform taxonomic classification of your reads, **Kaiju's** database index needs to be built from the reference protein database. A custom version of **Kaiju** with automatic download and build of the **MMP Mar databases** is available here.

Even a relative small database such as the **MMP Mar databases** generates large files when they are indexed (The indexing of the content is absolutely required in order to use a database). When **MMP Mar databases** are indexed properly, it requires 24 GB of space. You can choose to run **Kaiju** on your own data (it will take around 20 minutes), or we have pre-runned **Kaiju** with exactly the same dataset as you are using. The output from the prerun is found here `/practical/3/prerun`

I] **Alterntive 1:** Create a directory named `kaiju` in `/practical/3/` and make symbolic links to the FASTQ files in `/practical/2/rawdata/` in the new directory.

**Alterntive 2:** Move to the directory `/practical/3/prerun` where the pre-runned **Kaiju** output is and view what is in the directory. It should be four files in this directory.

II] View the **Kaiju** run options (or arguments), type:

```
kaiju -h
```

III] **Alterntive 1:** Run **Kaiju** against the Mar database:

**Alterntive 2:** The **Kaiju** output was generated using the following command:

```
kaiju -t /net/software/databases/kaijudb/nodes.dmp -f /net/software/databases/kaiju
```

Can you explain what the different arguments are?

- -t /path/to/nodes.dmp
- -f /path/to/kaiju_db.fmi
- -z 2
- -x
- -s 75
- Greedy mode [-a] allows mismatches
- -e 5
- -o sample_kaiju.out
- -v

▶ **Solution** - Click to expand

**Kaiju** will print one line for each read or read pair. The verbose output contains tab-separated columns with information about the length or score of the best match used for classification, NCBI taxon identifier of the assigned taxon, accession numbers, etc. See below or try:

```
head sample\_kaiju\_mar.out:

C   M01337:114:000000000-B86J6:1:1101:15174:1832    818 170 818,226186, NP_811857.1
C   M01337:114:000000000-B86J6:1:1101:22128:1836    586416  71  586416, WP_03855852
```

**Krona Tools** require an input file with the full taxon path, like shown below:

```
91803   root    cellular organisms  Bacteria    FCB group   Bacteroidetes/Chlorobi
87964   root    cellular organisms  Bacteria    FCB group   Bacteroidetes/Chlorobi
```

IV] Convert the output file into the proper **Krona Tools** format using **kaiju2krona**:

```
kaiju2krona -t /net/software/databases/kaijudb/nodes.dmp -n /net/software/databases
```

V] The file `sample_kaiju_mar.krona` can then be imported into **Krona** and converted into an HTML file using Krona's **ktImportText** program:

```
ktImportText -o sample_kaiju_krona_mar.html sample_kaiju_mar.krona
```

VI] Open the HTML file you just generated in a web browser.

❓ How many percent of the mapped reads have been assign to bacteria?

▶ **Solution** - Click to expand

❓ What is the most abundant genera, and how many percent of the reads have mapped to this genera?

▶ **Solution** - Click to expand

❓ Locate the specie *Aliivibrio logei*. How many of the reads in your sample originate from this specie?

▶ **Solution** - Click to expand

The program **kaijuReport** can convert Kaiju's tab-separated output file into a summary report file for a given taxonomic rank, e.g., species.

VII] Print a report at species level with species that comprise at least 1 percent of the total reads. Name the output file containing the report `sample_kaiju_mar_species.summary` :

```
kaijuReport -t /net/software/databases/kaijudb/nodes.dmp -n /net/software/databases
```

❓ What does the `-r` and flag do?

▶ **Solution** - Click to expand

❓ What is the most abundant specie and how large fraction of the reads in your sample originate from this specie?

▶ **Solution** - Click to expand

❓ How many of the reads in your sample originate from *Allivibrio salmonicida*?

▶ **Solution** - Click to expand

❓ Look at the last line in the report, how many percent of the reads could not be classified (does not match enything in the database)?

▶ **Solution** - Click to expand

❓ How many percent of the reads could not be assign at species level?

▶ **Solution** - Click to expand

VIII] Create a report on genus level. Name the output file containing the report `sample_kaiju_mar_genus.summary` . Hint: Look at exercise VI]

▶ **Hint** - Click to expand

❓ How many percent of the reads could not be assign at genus level?

▶ **Solution** - Click to expand

❓ What is the most abundant genera, and how many percent of the reads have mapped to this genera?

▶ **Solution** - Click to expand

In the folder `/practical/3/prerun/` we have run the same rawdata files against the Refseq database. The **Kaiju** output file is named `sample_kaiju_refseq.out` .

IX] Move to this folder and create a report on species level from this **Kaiju** output. Name the output file containing the report `sample_kaiju_refseq_species.summary` . Hint: Look at exercise VI]

❓ What are the major differences when comparing the same data against two different databases (if any)?

▶ **Solution** - Click to expand

X] Create a report on genus level and name the output report `sample_kaiju_refseq_genus.summary`

❓ What are the major differences when comparing the same data against two different databases (if any)?

▶ **Solution** - Click to expand

**Progress tracker**

Part 2 finished

# 3. Taxonomic classification using a K-mer based method


💡 **Kraken** is a system for ultrafast assignment of taxonomic labels to short DNA sequences using exact alignments of k-mers and a novel classification algorithm.

**Kraken's** default database contains just under 6 billion (6e9) distinct k-mers, and requires at least 160 GB of disk space. We will therefore use MiniKraken - Kraken run with a reduced database. The database is 4 GB and contains about 2.7% of kmers from the original database. You can read more here.

I] Move to the directory where you linked the sequence data to and view what is in the directory `/practical/3`

In order to classify reads with **Kraken**, you need to tell the program where the database is locate. We have downloaded the reduced database, formatted it and placed it here: `/net/software/databases/minikraken_20171013_4GB/`.

II] Run **Kraken** on the two FASTQ files and name the output `sample_kraken.out`. Remember that it is a paired end sample:

```
kraken --db /net/software/databases/minikraken_20171013_4GB/ --paired sample_R1.fas
```

❓ What does the --paired flag do?

▶ **Solution** - Click to expand

III] Look at the first lines in output file ( `head sample_kraken.out` ).

Each sequence classified by **Kraken** results in a single line of output. Output lines contain five tab-delimited fields; from left to right, they are:

- "C"/"U": one letter code indicating that the sequence was either classified or unclassified.
- The sequence ID, obtained from the FASTA/FASTQ header.
- The taxonomy ID Kraken used to label the sequence; this is 0 if the sequence is unclassified.
- The length of the sequence in bp.
- A space-delimited list indicating the LCA mapping of each k-mer in the sequence. For example, "562:13 561:4 A:31 0:1 562:3" would indicate that:
  - the first 13 k-mers mapped to taxonomy ID #562
  - the next 4 k-mers mapped to taxonomy ID #561
  - the next 31 k-mers contained an ambiguous nucleotide
  - the next k-mer was not in the database
  - the last 3 k-mers mapped to taxonomy ID #562

IV] In order to make a **Krona** chart of the **Kraken** output you need to remove some of the columns in the `sample_kraken.out` file:

```
cut -f2,3 sample_kraken.out > sample_kraken.krona.in
```

V] Make a **Krona** chart from the file `sample_kraken.krona.in` using Krona's **ktImportText** program:

```
ktImportTaxonomy sample_kraken.krona.in -o sample_kraken.krona.html
```

VI] Open the Krona chart from the **Kraken** analysis.

❓ What is the most abundant genera, and how of the reads have mapped to this genera?

▶ **Solution** - Click to expand

❓ Does any of the reads map to the specie *Aliivibrio logei*? How many of the reads in your sample originate from this specie?

▶ **Solution** - Click to expand

VII] Generate a taxonomic report of the **Kraken** output. NB! It is important that the database used must be the same as the one used to generate the output file:

```
kraken-report --db /net/software/databases/minikraken_20171013_4GB/ sample_kraken.o
```

The output of kraken-report is tab-delimited, with one line per taxon. The fields of the output, from left-to-right, are as follows:

- Percentage of reads covered by the clade rooted at this taxon
- Number of reads covered by the clade rooted at this taxon
- Number of reads assigned directly to this taxon
- A rank code, indicating (U)nclassified, (D)omain, (K)ingdom, (P)hylum, (C)lass, (O)rder, (F)amily, (G)enus, or (S)pecies. All other ranks are simply '-'.
- NCBI taxonomy ID
- indented scientific name

VIII] Open the Kraken report in a text editor

❓ How many percent of the reads were unclassified?

▶ **Solution** - Click to expand

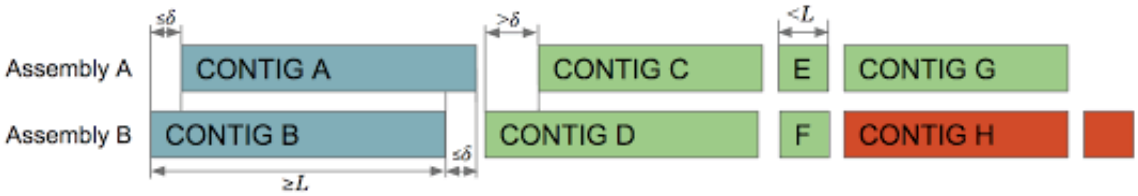
❓ How many percent of the reads belongs to the genus *Photobacterium*?

▶ **Solution** - Click to expand

**Progress tracker**

Part 3 finished

# 4. Taxonomic classification of contigs




Yesterday you assembled the sequence reads from this sample using **Megahit** and performed a comparison against several other assemblies. It is possible to perform taxonomic classification of contigs as well as on reads with the same tools. However, while the results from using reads will contain quantitative measures (eg. how many reads or percent of reads mapped to a specific taxon), the results from using contigs will not be quantitative - at least not using these methods. Using contigs, you will get an overview of what taxon are in you sample, not the relative abundences of them.

Just as for reads, the tools will try assign taxonomy to each contig and summarise how many contigs belong to each taxon. However, the tools do not differentiate between a long and a short contig. A taxon with many short contigs will appear highly abundant relative to a taxon with few large contigs, even though the total length of the large contigs is longer than the short.

I] Select one of the asseblies from yesterday and create a symbolic link to it in the directory `/practical/3`

You can reduse the number of false positive taxonomical assignments by removing very small contigs. Longer contigs will carry more sequenctial information that allows for more accurate sequence comparisons against databases, and producemore reliable taxonomical assignments.

II] Remove all contigs smaller than 300 bp from the assembly you choose. The following method using **awk** will only work on FASTA files with linearized sequences.

Linearize FASTA sequence:

```
sed -e 's/\(^>.*$\)/#\1#/' sample_metaspades_trim.fasta | tr -d "\r" | tr -d "\n" |
```

Remove contigs less than 300 bp:

```
awk '!/^>/ {next} {getline seq} length(seq) >= 300 {print $0 "\n" seq}' sample_meta
```

III] Count the number of contigs in both the original assembly and in the assembly with only contigs longer than 300 bp.

❓ How many contigs were removed?

▶ **Solution** - Click to expand

IV] Classify the contigs in the file with large contigs using **Kraken** and name the output

`sample_kraken_contigs.out` :

```
kraken --db /net/software/databases/minikraken_20171013_4GB/ sample_metaspades_larg
```

V] Generate a taxonomic report of the **Kraken** output:

```
kraken-report --db /net/software/databases/minikraken_20171013_4GB/ sample_kraken_c
```

❓ What is the most abundant genera, and how large fraction of the contigs have been assigned to this genera?

▶ **Solution** - Click to expand

❓ What was the most abundant genera when the taxonomic assignment was done on the reads using **Kraken**, and how large fraction of the reads was assigned to this genera?

▶ **Solution** - Click to expand

VI] Run Kaiju on the `sample_metaspades_large.fasta` Alternatively, we have pre-run **Kaiju** for you. The **Kaiju** output can be found here: `/practical/3/prerun/sample_kaiju_contigs.out`. The **Kaiju** output was generated using the following arguments:

```
kaiju -t /net/software/databases/kaijudb/nodes.dmp -f /net/software/databases/kaiju
```

VII] Generate a taxonomic report of the **Kaiju** output on genus level:

```
kaijuReport -t /net/software/databases/kaijudb/nodes.dmp -n /net/software/databases
```

❓ What is the most abundant genera, and how large fraction of the contigs have been assigned to this genera?

▶ **Solution** - Click to expand

❓ What was the most abundant genera when the taxonomic assignment was done on the reads using **Kaiju**, and how large fraction of the reads was assigned to this genera?

▶ **Solution** - Click to expand

**Progress tracker**

Complete

## That was the end of the this practical - Good job 👍