

---

# **Metagenomics Workshop Documentation**

***Release 1***

**MGnify team**

**Aug 09, 2022**



# CONTENTS

<b>1</b>	<b>Overview</b>	<b>3</b>
1.1	Assembling data . . . . .	3
1.2	MGnify Services . . . . .	7
1.3	MAG generation . . . . .	10
1.4	Quality control and filtering of the raw sequence files . . . . .	14
1.5	MGnify API data hunt . . . . .	24
1.6	License . . . . .	32



These are the materials that will be used for the “Metagenomics bioinformatics (Virtual)” workshop.



## OVERVIEW

Gain knowledge of the tools, processes and analysis approaches used in the field of metagenomics.

This course will cover the metagenomics data analysis workflow from the point of newly generated sequence data. Participants will explore the use of publicly available resources and tools to manage, share, analyse and interpret metagenomics data. The content will include issues of data quality control and how to submit to public repositories. While sessions will detail marker-gene and whole-genome shotgun (WGS) approaches; the primary focus will be on assembly-based approaches. Discussions will also explore considerations when assembling genome data, the analysis that can be carried out by MGnify on such datasets, and what downstream analysis options and tools are available

Contents:

## 1.1 Assembling data

- What constitutes a good assembly?
- How to estimate assembly quality
- Co-assembly



These instructions assume you are using the EBI Training Virtual Machines. If you're running this on your own computer instead, remember that some file paths will be different. In particular, `/home/training/` will be `/home/<yourusername>`.

### 1.1.1 Assembly and Co-assembly



**Learning Objectives** - in the following exercises you will learn how to perform a metagenomic assembly and to start some basic analysis of the output. Subsequently, we will demonstrate the application of co-assembly. Note, due to the complexity of metagenomics assembly, we will only be investigating very simple example datasets as these often take days of CPU time and 100s of GB of memory. Thus, do not think that there is an issue with the assemblies.

Once you have quality filtered your sequencing reads, you may want to perform *de novo* assembly in addition to, or as an alternative to a read-based analyses. The first step is to assemble your sequences into contigs. There are many tools available for this, such as MetaVelvet, metaSPAdes, IDBA-UD, MEGAHIT. We generally use metaSPAdes, as in most cases it yields the best contig size statistics (i.e. more contiguous assembly) and has been shown to be able to capture high degrees of community diversity (Vollmers, et al. PLOS One 2017). However, you should consider the pros and cons of different assemblers, which not only includes the accuracy of the assembly, but also their computational overhead. Compare these factors to what you have available. For example, very diverse samples with a lot of sequence data uses a lot of memory with SPAdes. In the following practicals we will demonstrate the use of metaSPAdes on a small sample and the use of MEGAHIT for performing co-assembly.



Let's first change to the working directory where we will be running the analyses:

```
cd /home/training/Data/Assembly/files
```



To run metaspades you would execute the following commands (but don't run it now!):

```
mkdir assembly
metaspades.py -t 4 --only-assembler -m 10 -1 reads/oral_human_example_1_splitaa_
kneaddata Paired_1.fastq -2 reads/oral_human_example_1_splitaa_kneaddata_Paired_2.
fastq -o assembly
```



Since the assembly process would take ~1h we are just going to analyse the output present in assembly.bak. Let's look at the contigs.fasta file.

For this, take the first 40 lines of the sequence and perform a blast search at NCBI (<https://blast.ncbi.nlm.nih.gov/Blast.cgi>, choose Nucleotide:Nucleotide from the set of options). Leave all other options as default on the search page. To select the first 40 lines of the assembly perform the following:

```
head -41 assembly.bak/contigs.fasta
```

Descriptions

Graphic Summary

Alignments

Taxonomy

Sequences producing significant alignments

Download Manage Columns Show 100

☒ select all
 19 sequences selected

	Description	Max Score	Total Score	Query Cover	E value	Per. Ident	Accession
<input checked="" type="checkbox"/>	<a href="#">Corynebacterium matruchotii strain NCTC10206 genome assembly, chromosome_1</a>	4183	4240	100%	0.0	98.13%	LR134504.1
<input checked="" type="checkbox"/>	<a href="#">Corynebacterium matruchotii strain ATCC 14266 chromosome</a>	4170	4227	100%	0.0	98.04%	CP050134.1
<input checked="" type="checkbox"/>	<a href="#">Corynebacterium atypicum strain R2070, complete genome</a>	333	333	32%	3e-86	74.84%	CP008944.1
<input checked="" type="checkbox"/>	<a href="#">Corynebacterium sp. HML98-0116 genome</a>	298	298	25%	1e-75	75.69%	CP017639.1
<input checked="" type="checkbox"/>	<a href="#">Corynebacterium testudinoris strain DSM 44614, complete genome</a>	296	296	30%	4e-75	74.22%	CP011545.1
<input checked="" type="checkbox"/>	<a href="#">Corynebacterium efficiens YS-314 DNA, complete genome</a>	283	283	36%	3e-71	72.86%	BA000035.2
<input checked="" type="checkbox"/>	<a href="#">Corynebacterium endometrit strain LMM-1653 chromosome, complete genome</a>	255	255	32%	6e-63	72.77%	CP038247.1
<input checked="" type="checkbox"/>	<a href="#">Corynebacterium callunae DSM 20147, complete genome</a>	156	156	36%	7e-33	70.37%	CP004354.1
<input checked="" type="checkbox"/>	<a href="#">Corynebacterium variabile DSM 44702, complete genome</a>	117	117	19%	3e-21	72.01%	CP002917.1
<input checked="" type="checkbox"/>	<a href="#">Rhodococcus sp. DMU1 plasmid unnamed</a>	89.8	89.8	4%	7e-13	81.03%	CP050953.1
<input checked="" type="checkbox"/>	<a href="#">Rhodococcus rhodochrous strain ATCC BAA870 chromosome, complete genome</a>	62.1	62.1	4%	2e-04	76.72%	CP032675.1
<input checked="" type="checkbox"/>	<a href="#">Rhodococcus pyridinivorans strain YF3 chromosome, complete genome</a>	62.1	62.1	4%	2e-04	76.72%	CP040719.1
<input checked="" type="checkbox"/>	<a href="#">Rhodococcus biohervivorans strain TQ9 chromosome, complete genome</a>	62.1	62.1	4%	2e-04	76.72%	CP022208.1
<input checked="" type="checkbox"/>	<a href="#">Rhodococcus pyridinivorans strain GF3, complete genome</a>	62.1	62.1	4%	2e-04	76.72%	CP022915.1
<input checked="" type="checkbox"/>	<a href="#">Rhodococcus sp. ZG, complete genome</a>	62.1	62.1	4%	2e-04	76.72%	CP018063.1
<input checked="" type="checkbox"/>	<a href="#">Rhodococcus sp. c52, complete genome</a>	62.1	62.1	4%	2e-04	76.72%	CP016819.1
<input checked="" type="checkbox"/>	<a href="#">Rhodococcus pyridinivorans SB3094, complete genome</a>	62.1	62.1	4%	2e-04	76.72%	CP006996.1
<input checked="" type="checkbox"/>	<a href="#">Salicicoccus sp. JOR-1 chromosome, complete genome</a>	54.7	54.7	1%	0.025	100.00%	CP042241.1
<input checked="" type="checkbox"/>	<a href="#">Rubrobacter xylanophilus DSM 9941, complete genome</a>	54.7	54.7	1%	0.025	100.00%	CP000386.1



Which species do you think this sequence may be coming from? Does this make sense as a human oral bacteria? Are you surprised by this result at all?



Now let us consider some statistics about the entire assembly

```
assembly_stats assembly.bak/scaffolds.fasta
```



This will output two simple tables in JSON format, but it is fairly simple to read. There is a section that corresponds



to the scaffolds in the assembly and a section that corresponds to the contigs.



What is the length of longest and shortest contigs?



What is the N50 of the assembly? Given that the input sequences were ~150bp long paired-end sequences, what does this tell you about the assembly?



N50 is a measure to describe the quality of assembled genomes that are fragmented in contigs of different length. We can apply this with some caution to metagenomes, where we can use it to crudely assess the contig length that covers 50% of the total assembly. Essentially the longer the better, but this only makes sense when thinking about alike metagenomes. Note, N10 is the minimum contig length to cover 10 percent of the metagenome. N90 is the minimum contig length to cover 90 percent of the metagenome.



Bandage (a Bioinformatics Application for Navigating De novo Assembly Graphs Easily), is a program that creates interactive visualisations of assembly graphs. They can be useful for finding sections of the graph, such as rRNA, or to try to find parts of a genome. Note, you can install Bandage on your local system. With Bandage, you can zoom and pan around the graph and search for sequences, plus much more. The following guide allows you to look at the assembly graph. Normally, I would recommend looking at the 'assembly\_graph.fastg', but our assembly is quite fragmented, so we will load up the assembly\_graph\_after\_simplification.gfa.



At the terminal, type

Bandage

In the the Bandage GUI perform the following

Select File -> Load graph

Navigate to Home -> Data -> Assembly -> files -> assembly.bak and open the file assembly\_graph\_after\_simplification.gfa

Once loaded, you need to draw the graph. To do so, under the "Graph drawing" panel on the left side perform the following:

Set Scope to 'Entire graph'

Then click on Draw graph



Use the sliders in the main panel to move around and look at each distinct part of the assembly graph.



Can you find any large, complex parts of the graph? If so, what do they look like.



In this particular sample, we believe that strains related to the species *Rothia dentocariosa*, a Gram-positive, round- to rod-shaped bacteria that is part of the normal community of microbes residing in the mouth and respiratory tract, should be present in our sample. While this is a tiny dataset, let's try to see if there is evidence for this genome. To do so, we will search the *R. dentocariosa* genome against the assembly graph.

To do so, go to the 'BLAST' panel on the left side of the GUI.

Step 1 - Select 'Create/view BLAST search', this will open a new window

Step 2 - Select 'build Blast database'

Step 3 - Load from FASTA file. Navigate to the genome folder: Home -> Data -> Assembly -> files -> genome and select GCA\_000164695.fasta

Step 4 - Modify the BLAST filters to 95% identity

Step 5 - Run BLAST

Step 6 - Close this window

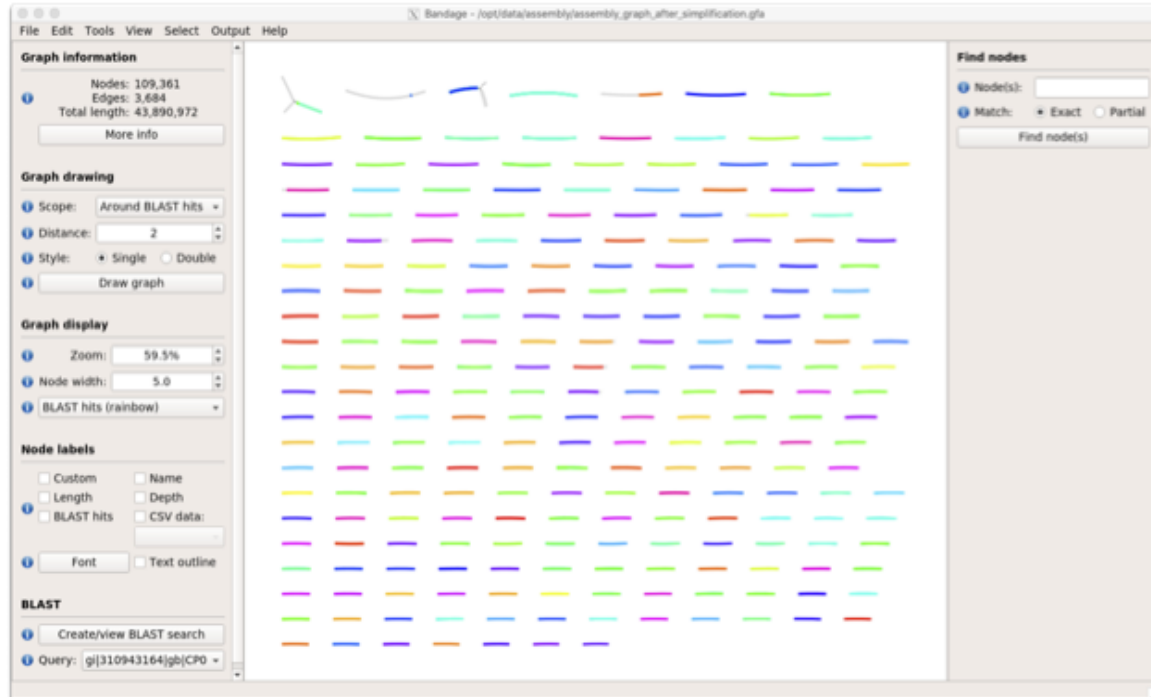
To visualise just these hits, go back to “Graph drawing” panel.

Set Scope to ‘Around BLAST hits’

Set Distance 2

The click on ‘Draw graph’

You should then see something like this:



In the following steps of this exercise, we will look at performing co-assembly of multiple datasets. Each should take about 15-20 min. In case you do not manage to finish these on time, the directory **coassembly.bak** contains all the expected results.



First, we need to make sure the output directories we are going to create do not already exist (MEGAHIT cannot overwrite existing directories). Run:

```
rm -rf coassembly/assembly*
```



Then, perform the coassemblies with MEGAHIT, as follows:

```
megahit -1 reads/oral_human_example_1_splitac_kneaddata_paired_1.fastq -2 reads/oral_
human_example_1_splitac_kneaddata_paired_2.fastq -o coassembly/assembly1 -t 4 --k-
list 23,51,77
```

```
megahit -1 reads/oral_human_example_1_splitac_kneaddata_paired_1.fastq,reads/oral_human_
example_1_splitab_kneaddata_paired_1.fastq -2 reads/oral_human_example_1_splitac_
```

(continues on next page)

(continued from previous page)

```
↪kneaddatapaired_2.fastq,reads/oral_human_example_1_splitab_kneaddatapaired_2.fastq -
↪o coassembly/assembly2 -t 4 --k-list 23,51,77
```

```
megahit -1 reads/oral_human_example_1_splitab_kneaddatapaired_1.fastq,reads/oral_human_
↪example_1_splitac_kneaddatapaired_1.fastq,reads/oral_human_example_1_splitaa_
↪kneaddatapaired_1.fastq -2 reads/oral_human_example_1_splitab_kneaddatapaired_2.
↪fastq,reads/oral_human_example_1_splitac_kneaddatapaired_2.fastq,reads/oral_human_
↪example_1_splitaa_kneaddatapaired_2.fastq -o coassembly/assembly3 -t 4 --k-list 23,51,
↪77
```



You should now have three different assemblies, let us compare the results.

```
assembly_stats coassembly/assembly1/final.contigs.fa
assembly_stats coassembly/assembly2/final.contigs.fa
assembly_stats coassembly/assembly3/final.contigs.fa
```



How do these assemblies differ to the one generated previously with metaSPAdes? Which one do you think is best?

## 1.2 MGnify Services

In the presentation we will cover:

- Overview of MGnify annotation pipeline
- Taxonomic assignment
- Functional characterisation
- Pathways/systems
- Other MGnify services

### 1.2.1 MGnify hands-on exercises

For this session we will look at some of the data and analyses that are available from MGnify. We will navigate the resource, try out different ways to search for interesting samples/studies, and then investigate the analysis results that are available.

#### Browsing MGnify

From the MGnify front page (<https://www.ebi.ac.uk/metagenomics/>) you can see various options to browse the data. There are quick links to the various data-types (e.g. amplicon, assembly, metagenomes, etc) we support, as well as a subset of the biomes that the data covers.




Click on the “wastewater” biome icon.





How many studies does MGnify hold that relate to wastewater?





How many samples does that relate to?


 From the sample list, filter for sample “ERS1215575”, select it and take a look at the metadata available.


 Do you know the exact location of where the sample was taken?

 What are the lat/long co-ordinates?

 Follow the link to the BioSamples record, can you find any more information about the location of the sample?


 From the tabs in the header bar, select **Text search**, and then select **Samples** below the search box. There are a number of metadata fields available to allow you to filter for a sample of interest to you. Not all are relevant to all samples. Within the hierarchy of biomes, navigate to environmental>aquatic>lentic. You should see 57 samples. Now select the depth filter.


 How many lentic samples have depth data associated with them?


 Using the sliders, can you identify a sample of a lentic water system from a depth between 25-50m?


### MGnify assembly analysis


Now we will look at some assembly data that has been analysed by MGnify.


 Search for **MGYS00003598**, and go to this study page. This is a large study where MGnify have assembled the raw reads from an existing public study. The list of assemblies is shown at the bottom of the study page.

 How many assemblies are included in this study?

 Click on the 2nd analysis link in the list **MGYA00510849**. You could alternatively search for this accession using the text search options. Have a look at the information within the **Quality control** tab.


 How many contigs are included in this analysis?


 What length is the longest contig in this dataset?


 Click on the **Taxonomic analysis** tab and examine the phylum composition in the graphs and the krona plot.

 What proportion of the total LSU rRNA predictions are eukaryotic?

 What proportion of the bacterial predictions are proteobacteria?

 Click on the **Functional analysis** tab. The top part of this page shows a sequence feature summary, showing the number of contigs with predicted coding sequences (pCDS), the number of pCDS with InterPro matches, etc.

 How many predicted coding sequences (pCDS) are in the assembly?

 How many pCDS have InterProScan hits?



Scroll down the page to the InterPro match summary section.



How many different InterPro entries are matched by the pCDS?



Why is this figure different to the number of pCDS that have InterProScan hits?



Click on the **GO Terms** sub-tab. This shows a summary of the most common GO terms annotated to the pCDS as both bar charts, and pie charts.



What are the top 3 biological process terms predicted for the pCDS from this assembly?



Have a look at the information in the **Pfam** and **KO** (KEGG orthologue) sub-tabs.



Click on the **Pathways/Systems** tab. Have a look at the data reported in the 3 sub-tabs: KEGG Module, Genome Properties, and antiSMASH.



How many KEGG modules are reported for this assembly?



How many of these are 100% complete (i.e. all of the constituent KOs are found)?



How many Genome Properties of the category **DNA handling**, are found within this assembly?



What is the most common class of biosynthetic gene cluster found in this assembly?



How many non-ribosomal peptide synthetase gene clusters are identified by antiSMASH in this assembly?



Click on the **Contig Viewer** tab. Load the data for the 4th contig in the list by clicking on the contig name (ERZ501066.4-NODE-4-length-276957-cov-33.799655). This contig will now be loaded into the viewer.



How long is this contig?



The longest pCDS in the contig appears to start at 202339. What protein is coded for?



Looking at the antiSMASH annotations, where within the contig do any transport-related genes fall?



Zoom into that region to see the predicted regions in more detail. Have a look at the information about the various transport-related genes.



What region of the contig is predicted to code for a major facilitator transporter?



There are lots of different visualisation options available within the contig viewer. Take some time now to investigate the various options, and play about with it by looking at a few different contigs and the anotations they contain.

## MGnify sequence search

Now we will have a look at the database of proteins identified by MGnify.



Click on “Sequence search” from the tabs at the top of the page.

This will open a HMMER search page specific for MGnify. (For more information about the HMMER suite of tools see the HMMER website <https://www.ebi.ac.uk/Tools/hmmer/>)



Copy and paste the protein sequence below into the sequence search box at the top of the page, and click “submit”.

```
GEFWHWTNLLHFILVGLAGGMAFLTALLHLKGHPARRYTLWALGLIALDLFVLWAESPARFRFTHV
WLFLSFHPTSPIWWSWGLALSVSAGLLYLKGKPSKPLAWGLLAFSLVALAYPGMALAVNLNRPLWN  AL-
LAGLFPALTALVLGLGVAVLMKSSWALYPLRILLGASLFLAFLYPFTLTLEARGHLWEEGGVLYGL  FLALGL-
GAFGKESLAPWAAFLAAAGLRALLVAVGQWQG
```



How many query results are significant? (i.e. above the red cut-off line)



Click on the “Customise” button at the top right of the results table, and select to make “Run and sample IDs” column visible and click “Update”. Have a look at the sample data for some of the runs listed in the results (for example the top match result).



Looking at the samples included in the significant results, does it make sense that the example sequence was from this protein family? IPR032796 - Polysulfide reductase.

## 1.3 MAG generation

- Generation of metagenome assembled genomes (MAGs) from assemblies
- Assessment of quality (MIGMAGs)
- Taxonomic assignment



These instructions assume you are using the EBI Training Virtual Machines. If you’re running this on your own computer instead, remember that some file paths will be different. In particular, `/home/training/` will be `/home/<yourusername>`.

### 1.3.1 Prerequisites

For this tutorial you will need to first start the docker container by running:

```
sudo docker run --rm -it -v /home/training/Data/Binning:/opt/data microbiomeinformatics/
↪mgnify-ebi-2020-binning
```

---

**Note:** It’s possible that the docker image is not available in dockerhub. In that case you can build the container using the [Dockerfile](#)

To build the container, download the Dockerfile and run “`docker build -t microbiomeinformatics/mgnify-ebi-2020-binning .`” in the folder that contains the Dockerfile.

---

password: training

### 1.3.2 Generating metagenome assembled genomes

**i** Learning Objectives - in the following exercises you will learn how to bin an assembly, assess the quality of this assembly with checkM and then visualise a placement of these genomes within a reference tree.

**i** As with the assembly process, there are many software tools available for binning metagenome assemblies. Examples include, but are not limited to:

MaxBin: <https://sourceforge.net/projects/maxbin/>

CONCOCT: <https://github.com/BinPro/CONCOCT>

COCACOLA: <https://github.com/younglululu/COCACOLA>

MetaBAT: <https://bitbucket.org/berkeleylab/metabat>

There is no clear winner between these tools, so the best is to experiment and compare a few different ones to determine which works best for your dataset. For this exercise we will be using **MetaBAT** (specifically, MetaBAT2). The way in which MetaBAT bins contigs together is summarised in Figure 1.

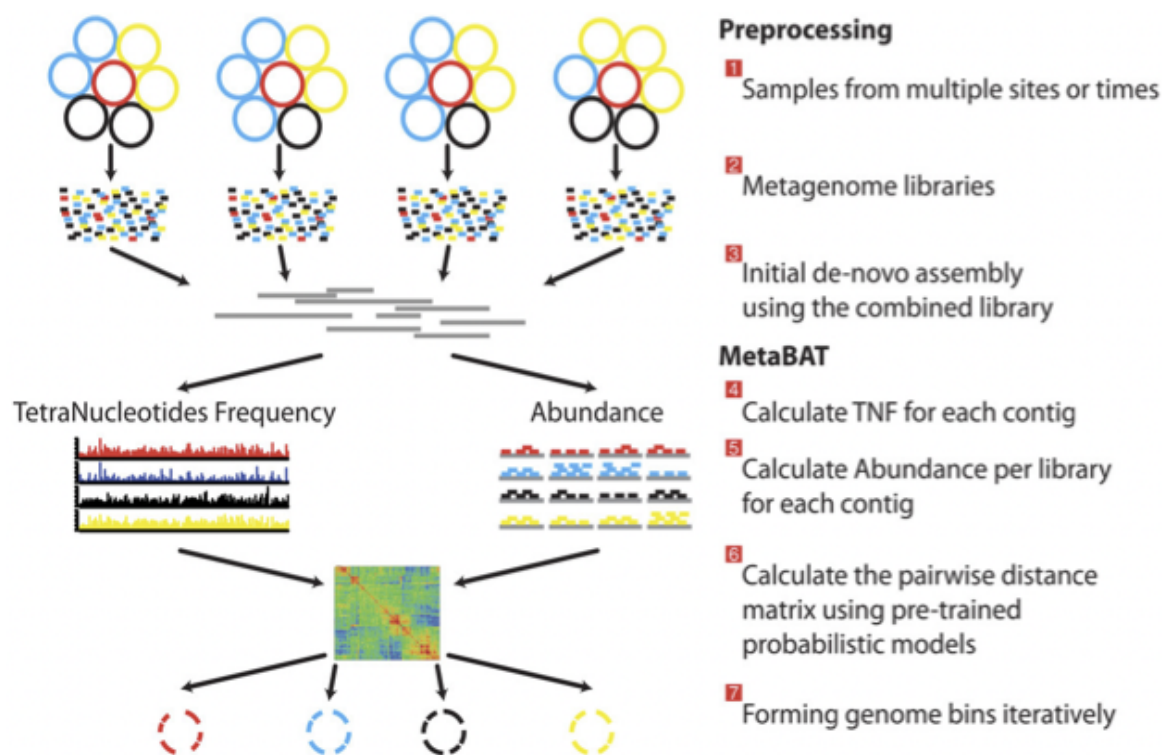


Figure 1. MetaBAT workflow (Kang, et al. *PeerJ* 2015).

**i** Prior to running MetaBAT, we need to generate coverage statistics by mapping reads to the contigs. To do this, we can use bwa (<http://bio-bwa.sourceforge.net/>) and then the samtools software (<http://www.htslib.org>) to reformat the output. Again, this can take some time, so we have run it in advance. To repeat the process, you would run the following commands:

```
# index the contigs file that was produced by metaSPAdes:
bwa index contigs.fasta
```

(continues on next page)

(continued from previous page)

```
# map the original reads to the contigs:
bwa mem contigs.fasta ERR011322_1.fastq ERR011322_2.fastq > input.fastq.sam

# reformat the file with samtools:
samtools view -Sbu input.fastq.sam > junk
samtools sort junk input.fastq.sam
```

We should now have the files we need for the rest of the process – the assemblies themselves (*contigs.fasta*) and a file from which we can generate the coverage stats (*input.fastq.sam.bam*).

## Running MetaBAT



Create a subdirectory where files will be output:

```
cd /opt/data/assemblies/
mkdir contigs.fasta.metabat-bins2000
```

In this case, the directory might already be part of your VM, so do not worry if you get an error saying the directory already exists. You can move on to the next step.



Run the following command to produce a *contigs.fasta.depth.txt* file, summarising the output depth for use with MetaBAT:

```
jgi_summarize_bam_contig_depths --outputDepth contigs.fasta.depth.txt input.fastq.sam.bam
```



Now you can run MetaBAT as:

```
metabat2 --inFile contigs.fasta --outFile contigs.fasta.metabat-bins2000/bin --abdFile_
↪ contigs.fasta.depth.txt --minContig 2000
```



Once the binning process is complete, each bin will be grouped into a multi-fasta file with a name structure of **bin.[0-9].fa**.



Inspect the output of the binning process.

```
ls contigs.fasta.metabat-bins2000/bin*
```



How many bins did the process produce?



How many sequences are in each bin?

Obviously, not all bins will have the same level of accuracy since some might represent a very small fraction of a potential species present in your dataset. To further assess the quality of the bins we will use **CheckM** (<https://github.com/Ecogenomics/CheckM/wiki>).

## Running CheckM



**CheckM** has its own reference database of single-copy marker genes. Essentially, based on the proportion of these markers detected in the bin, the number of copies of each and how different they are, it will determine the level of **completeness**, **contamination** and **strain heterogeneity** of the predicted genome.



Before we start, we need to configure checkM.



```
checkm data setRoot /opt/data/checkm_data
```

This program has some handy tools not only for quality control, but also for taxonomic classification, assessing coverage, building a phylogenetic tree, etc. The most relevant ones for this exercise are wrapped into the **lineage\_wf** workflow.

Now run CheckM with the following command:

```
checkm lineage_wf -x fa contigs.fasta.metabat-bins2000 checkm_output --tab_table -f MAGs_
→checkm.tab --reduced_tree -t 4
```

Due to memory constraints (< 40 GB), we have added the option **--reduced\_tree** to build the phylogeny with a reduced number of reference genomes.

Once the **lineage\_wf** analysis is done, the reference tree can be found in **checkm\_output/storage/tree/concatenated.tre**.

Additionally, you will have the taxonomic assignment and quality assessment of each bin in the file **MAGs\_checkm.tab** with the corresponding level of **completeness**, **contamination** and **strain heterogeneity** (Fig. 2). A quick way to infer the overall quality of the bin is to calculate the level of **(completeness - 5\*contamination)**. You should be aiming for an overall score of at least **70-80%**.

You can inspect the CheckM output with:

```
cat MAGs_checkm.tab
```

Bin Id	Marker lineage	# genomes	# markers	# marker sets	0	1	2	3	4	5+	Completeness	Contamination	Strain heterogeneity
bin.1	f_Lachnospiraceae (UID1286)	57	420	207	27	383	10	0	0	0	95.94	2.42	50
bin.2	k_Bacteria (UID203)	5449	104	58	5	22	28	44	5	0	97.39	163.71	42.63
bin.3	o_Clostridiales (UID1212)	172	263	149	9	253	1	0	0	0	96.64	0.67	100
bin.4	o_Clostridiales (UID1212)	172	263	149	65	197	1	0	0	0	75.03	0.67	0
bin.5	k_Bacteria (UID2565)	2921	152	93	13	139	0	0	0	0	88.17	0	0

Figure 2. Example output of CheckM

Before we can visualize and plot the tree we will need to convert the reference ID names used by CheckM to taxon names. We have already prepared a mapping file for renaming the tree (**rename\_list.tab**). We can then do this easily with the **newick utilities** ([http://cegg.unige.ch/newick\\_utils](http://cegg.unige.ch/newick_utils)).

To do this, run the following command:

```
nw_rename checkm_output/storage/tree/concatenated.tre rename_list.tab > renamed.tree
```

### Visualising the phylogenetic tree

We will now plot and visualize the tree we have produced. A quick and user- friendly way to do this is to use the web-based **interactive Tree of Life (iTOL)**: <http://itol.embl.de/index.shtml>

**iTOL** only takes in newick formatted trees, so we need to quickly reformat the tree with **FigTree** (<http://tree.bio.ed.ac.uk/software/figtree/>).

In order to open **FigTree** navigate to: **Home -> Data -> Binning -> FigTree\_v1.4.4 -> lib -> figtree.jar**



Open the **renamed.tree** file with **FigTree** (**File -> Open**) and then select from the toolbar **File -> Export Trees**. In the **Tree file format** select **Newick** and export the file as **renamed.nwk** (or choose a name you will recognise if you plan to use the shared account described below).



To use **iTOL** you will need a user account. For the purpose of this tutorial we have already created one for you with an example tree. The login is as follows:

User: *EBI\_training*

Password: *EBI\_training*

After you login, just click on **My Trees** in the toolbar at the top and select

**IBD\_checkm.nwk** from the **Imported trees** workspace.

Alternatively, if you want to create your own account and plot the tree yourself follow these steps:

- 1) After you have created and logged in to your account go to **My Trees**
- 2) From there select **Upload tree files** and upload the tree you exported from **FigTree**
- 3) Once uploaded, click the tree name to visualize the plot
- 4) To colour the clades and the outside circle according to the phylum of each strain, drag and drop the files **iTOL\_clades.txt** and **iTOL\_ocircles.txt** present in `/home/training/Data/Binning/iTOL_Files/` into the browser window

Once that is done, all the reference genomes used by **CheckM** will be coloured according to their phylum name, while all the other ones left blank correspond to the **target genomes** we placed in the tree. Highlighting each tip of the phylogeny will let you see the whole taxon/sample name. Feel free to play around with the plot.



Does the CheckM taxonomic classification make sense? What about the unknowns? What is their most likely taxon?

## 1.4 Quality control and filtering of the raw sequence files



These instructions assume you are using the EBI Training Virtual Machines. If you're running this on your own computer instead, remember that some file paths will be different. In particular, `/home/training/` will be `/home/<yourusername>`.

### 1.4.1 Prerequisites

For this tutorial you will need to move into the working directory. All the required files should be here, or can be downloaded using the tarball from [http://ftp.ebi.ac.uk/pub/databases/metagenomics/mgnify\\_courses/ebi\\_2020/](http://ftp.ebi.ac.uk/pub/databases/metagenomics/mgnify_courses/ebi_2020/)

```
cd /home/training/Data/Quality/files
chmod -R 777 /home/training/Data/Quality
export DATADIR=/home/training/Data/Quality/files
xhost +
```

Finally, start the docker container in the following way:

```
docker run --rm -it -e DISPLAY=$DISPLAY -v $DATADIR:/opt/data -v /tmp/.X11-unix:/tmp/.X11-unix:rw -e DISPLAY=unix$DISPLAY microbiomeinformatics/mgnify-ebi-2020-qc-assembly
```

---

**Note:** It's possible that the docker image is not available in dockerhub. In that case you can build the container using the [Dockerfile](#)

To build the container, download the Dockerfile and run “`docker build -t microbiomeinformatics/mgnify-ebi-2020-qc-assembly .`” in the folder that contains the Dockerfile.

---

## 1.4.2 Quality control and filtering of the raw sequence files



Learning Objectives - in the following exercises you will learn how to check on the quality of short read sequences: identify the presence of adaptor sequences, remove both adaptors and low quality sequences. You will also learn how to construct a reference database for host decontamination.



First go to your working area, the data that you downloaded has been mounted in `/opt/data` in the docker container.

```
cd /opt/data
ls
```



Here you should see the same contents as you had in the working directory. As we write into this directory, we should be able to see this from inside the container, and on the filesystem of the computer running this container. We will use this to our advantage as we go through this practical. Unless stated otherwise all of the following commands should be executed in the terminal running the Docker container.



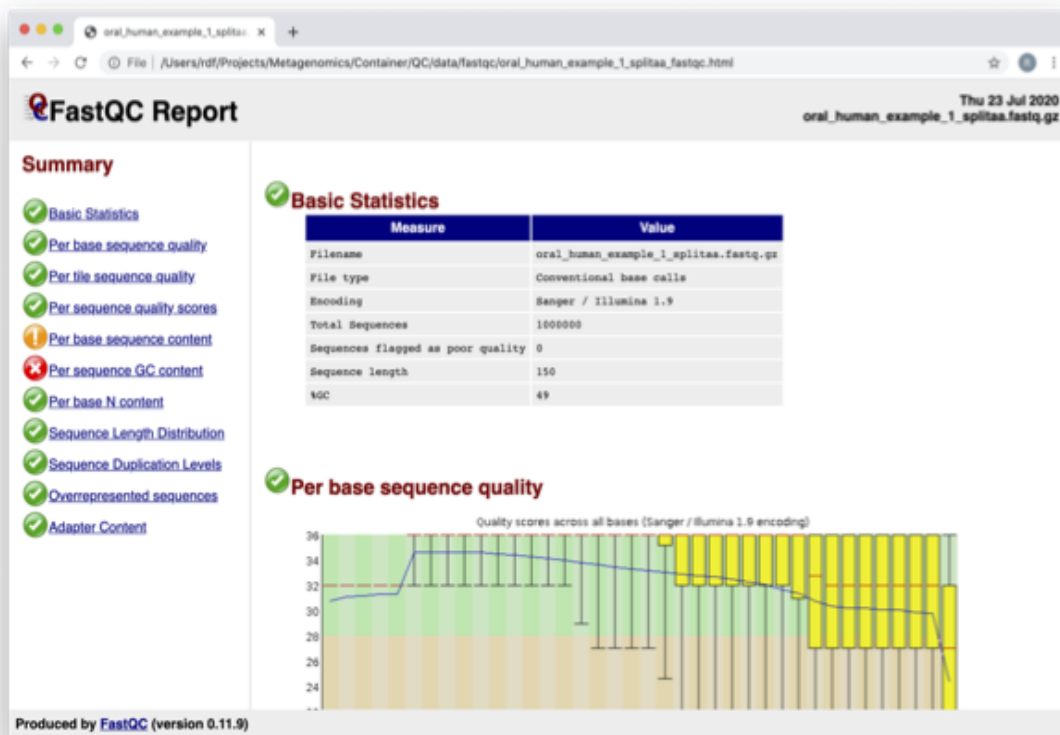
Generate a directory of the fastqc results

```
cd /opt/data
mkdir fastqc_results
fastqc oral_human_example_1_splitaa.fastq.gz
fastqc oral_human_example_2_splitaa.fastq.gz
mv /opt/data/*.zip /opt/data/fastqc_results
mv /opt/data/*.html /opt/data/fastqc_results
```



Now on your **local** computer, go to the browser, and File -> Open File. Use the file navigator to select the following file

`/home/training/Data/Quality/files/fastqc_results/oral_human_example_1_splitaa_fastqc.html`



Spend some time looking at the ‘Per base sequence quality’.



For each position a BoxWhisker type plot is drawn. The elements of the plot are as follows:

- The central red line is the median value
- The yellow box represents the inter-quartile range (25-75%)
- The upper and lower whiskers represent the 10% and 90% points
- The blue line represents the mean quality

The y-axis on the graph shows the quality scores. The higher the score the better the base call. The background of the graph divides the y axis into very good quality calls (green), calls of reasonable quality (orange), and calls of poor quality (red). The quality of calls on most platforms will degrade as the run progresses, so it is common to see base calls falling into the orange area towards the end of a read.



What does this tell you about your sequence data? When do the errors start?

In the pre-processed files we see two warnings, as shown on the left side of the report. Navigate to the “Per bases sequence content”



Q At around 15-19 nucleotides, the DNA composition becomes very even, however, at the 5' end of the sequence there are distinct differences. Why do you think that is?

I Open up the FastQC report corresponding to the reversed reads.

Q Are there any significant differences between the forward and reverse files?

For more information on the FastQC report, please consult the 'Documentation' available from this site: <https://www.bioinformatics.babraham.ac.uk/projects/fastqc/>

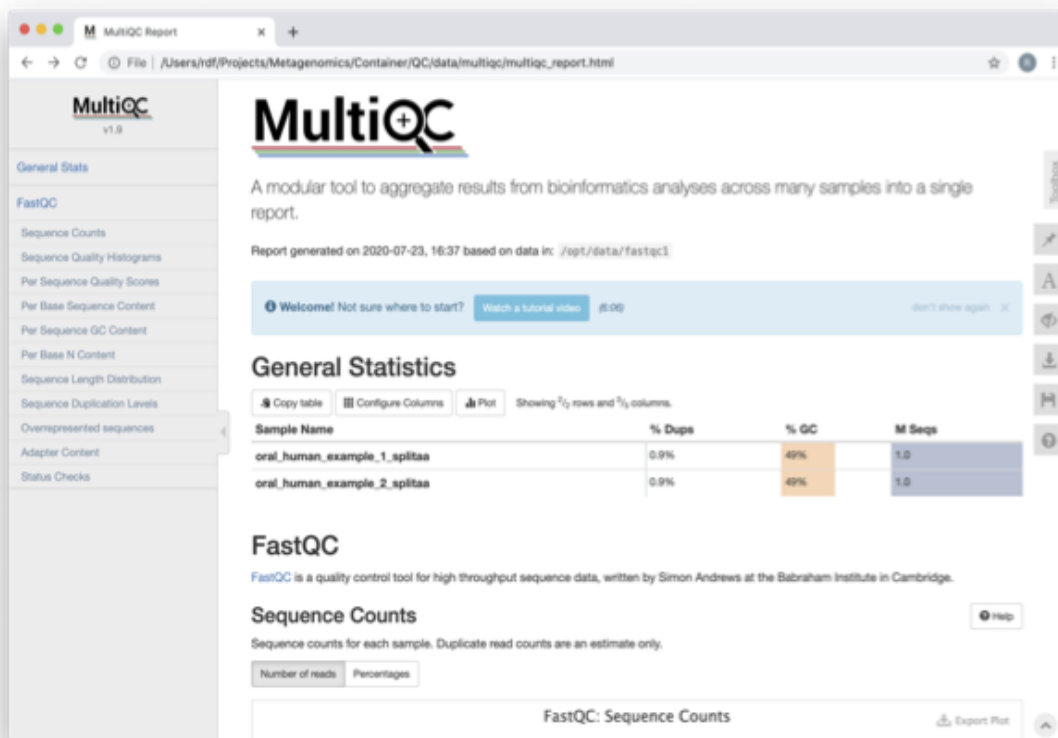
I We are currently only looking at two files but often we want to look at many files. The tool multiqc aggregates the FastQC results across many samples and creates a single report for easy comparison. Here we will demonstrate the use of this tool

```
cd /opt/data
mkdir multiqc_results
multiqc fastqc_results -o multiqc_results
```

In this case, we provide the folder containing the fastqc results to multiqc and the -o allows us to set the output directory for this summarised report.

I Now on your **local** computer, open the summary report from MultiQC. To do so, go to your browser, and use File -> Open File. Use the file navigator to select the following file

/home/training/Data/Quality/files/multiqc\_results/multiqc\_report.html



Scroll down through the report. The sequence quality histograms show the following results from each file as two separate lines. The ‘Status Checks’ show a matrix of which samples passed check and which ones have problems.

What fraction of reads are duplicates?

So, far we have looked at the raw files and assessed their content, but we have not done anything about removing duplicates, sequences with low quality scores or removal of the adaptors. So, lets start this process. The first step in the process is to make a database relevant for decontaminating the sample. It is always good to routinely screen for human DNA (which may come from the host and/or staff performing the experiment). However, if the sample is say from mouse, you would want to download the the mouse genome.

In the following exercise, we are going to use two “genomes” already downloaded for you in the decontamination folder. To make this tutorial quicker and smaller in terms of file sizes, we are going to use PhiX (a common spike in) and just chromosome 10 from human.

```
cd /opt/data/decontamination
```

For the next step we need one file, so we want to merge the two different fasta files. This is simply done using the command line tool cat.

```
cat phix.fasta GRCh38_chr10.fasta > GRCh38_phix.fasta
```

Now we need to build a bowtie index for them:

```
bowtie2-build GRCh38_phix.fasta GRCh38_phix.index
```



It is possible to automatically download a pre-indexed human genome in Bowtie2 format using the following command (but do not do this now, as this will take a while to download):

```
kneaddata_database --download human_genome bowtie2
```



Now we are going to use the *GRCh38\_phix* database and clean-up our raw sequences. *kneaddata* is a helpful wrapper script for a number of pre-processing tools, including Bowtie2 to screen out contaminant sequences, and Trimmomatic to exclude low-quality sequences. We also have written wrapper scripts to run these tools (see below), but using *kneaddata* allows for more flexibility in options.

```
cd /opt/data/  
mkdir clean
```

We now need to uncompress the fastq files.

```
gunzip -c oral_human_example_2_splitaa.fastq.gz > oral_human_example_2_splitaa.fastq  
gunzip -c oral_human_example_1_splitaa.fastq.gz > oral_human_example_1_splitaa.fastq  
  
kneaddata --remove-intermediate-output -t 2 --input oral_human_example_1_splitaa.fastq --  
→input oral_human_example_2_splitaa.fastq --output /opt/data/clean --reference-db /opt/  
→data/decontamination/GRCh38_phix.index --bowtie2-options "--very-sensitive --dovetail"  
→--trimmomatic-options "SLIDINGWINDOW:4:20 MINLEN:50"
```



The options above are:

- \* **-input**, Input FASTQ file. This option is given twice as we have paired-end data.
- \* **-output**, Output directory.
- \* **-reference-db**, Path to bowtie2 database for decontamination.
- \* **-t**, # Number of threads to use (2 in this case).
- \* **-trimmomatic-options**, Options for Trimmomatic to use, in quotations ("SLIDINGWINDOW:4:20 MINLEN:50" in this case). See the Trimmomatic website for more options.
- \* **-bowtie2-options**, Options for bowtie2 to use, in quotations. The options "--very-sensitive" and "--dovetail" set the alignment parameters to be very sensitive and sets cases where mates extend past each other to be concordant (i.e. they will be called as contaminants and be excluded).
- \* **-remove-intermediate-output**, Intermediate files, including large FASTQs, will be removed.

**Kneaddata generates multiple outputs in the "clean" directory, containing different 4 different files for each read.**



Using what you have learned previously, generate a fastqc report for each of the *oral\_human\_example\_1\_splitaa\_kneaddata\_paired* files. Do this within the clean directory.

```
cd /opt/data/clean  
mkdir fastqc_final  
<you construct the commands>  
mv /opt/data/clean/*.zip /opt/data/clean/fastqc_final  
mv /opt/data/clean/*.html /opt/data/clean/fastqc_final
```

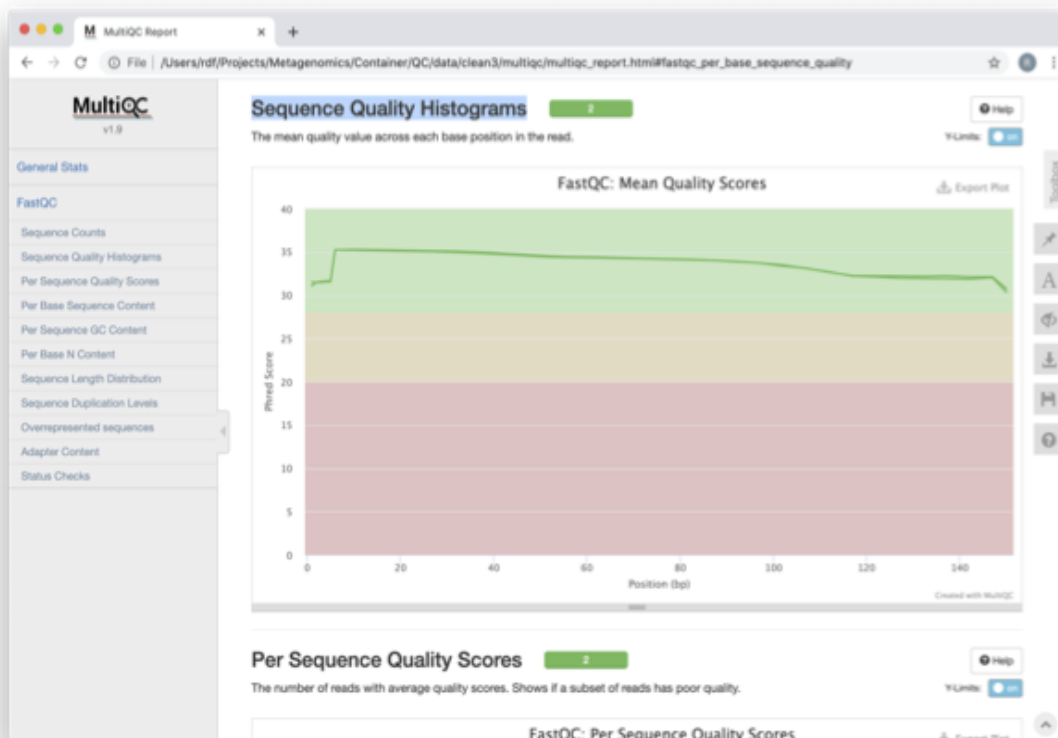


Also generate a multiqc report and look at the sequence quality histograms.

```
cd /opt/data/clean
mkdir multiqc_final
<you construct the command>
```



View the multiQC report as before using your browser. You should see something like this:



Open the previous MultiQC report and see if they have improved?



Did sequences at the 5' end become uniform? Why might that be? Is there anything that suggests that adaptor sequences were found?



To generate a summary file of how the sequence were categorised by Kneaddata, run the following command.

```
cd /opt/data
kneaddata_read_count_table --input /opt/data/clean --output kneaddata_read_counts.txt
less kneaddata_read_counts.txt
```



What fraction of reads have been deemed to be contaminating?



The reads have now be decontaminated any can be uploaded to ENA, one of the INSDC members. It is beyond the scope of this course to include a tutorial on how to submit to ENA, but there is additional information available on how to do this in this Online Training guide provided by EMBL-EBI

<https://www.ebi.ac.uk/training/online/course/ebi-metagenomics-portal-submitting-metagenomics-da/considerations-submitting-metagenomic-data>



### 1.4.3 Assembly PhiX decontamination



Learning Objectives - in the following exercises you will generate a PhiX blast database, and run a blast search with a subset of assembled freshwater sediment metagenomic reads, to identify contamination.

PhiX, used in the previous section of this practical, is a small bacteriophage genome typically used as a calibration control in sequencing runs. Most library preparations will use PhiX at low concentrations, however it can still appear in the sequencing run. If not filtered out, PhiX can form small spurious contigs which could be incorrectly classified as diversity.



Generate the PhiX reference blast database

```
cd /opt/data/decontamination
makeblastdb -in phix.fasta -input_type fasta -dbtype nucl -parse_seqids -out phix_blastDB
```



Prepare the freshwater sediment example assembly file and search against the new blast database. This assembly file contains only a subset of the contigs for the purpose of this practical.

```
cd /opt/data
gunzip -c freshwater_sediment_contigs.fa.gz > freshwater_sediment_contigs.fa
blastn -query freshwater_sediment_contigs.fa -db decontamination/phix_blastDB -task_
megablast -word_size 28 -best_hit_overhang 0.1 -best_hit_score_edge 0.1 -dust yes -
-evalue 0.0001 -min_raw_gapped_score 100 -penalty -5 -soft_masking true -window_size_
100 -outfmt 6 -out freshwater_blast_out.txt
```



The blast options are:

- \* **-query**, Input assembly fasta file.
- \* **-out**, Output file
- \* **-db**, Path to blast database.
- \* **-task**, Search type -“megablast”, for very similar sequences (e.g, sequencing errors)
- \* **-word\_size**, Length of initial exact match



Add headers to the blast output and look at the contents of the final output file

```
cat blast_outfmt6.txt freshwater_blast_out.txt > freshwater_blast_out_headers.txt
less freshwater_blast_out_headers.txt
```



Are the hits significant?



What are the lengths of the matching contigs? We would typically filter metagenomic contigs at a length of 500bp. Would any PhiX contamination remain even after this filter?



Now that PhiX contamination was identified, it is important to remove these contigs from the assembly file before further analysis or upload to public archives.

## 1.4.4 Using Negative Controls



**Learning Objectives** - This exercise will look at the analysis of negative controls. You will assess the microbial diversity between a negative control and skin sample.

The images below show the taxonomic classification of two samples: a reagent negative control and a skin metagenomic sample. The skin sample is taken from the antecubital fossa - the elbow crease, which is moist and site of high microbial diversity. The classification was performed with kraken2. Kraken2 takes a while to run, so we have done this for you and plotted the results. An example of the command used to do this:

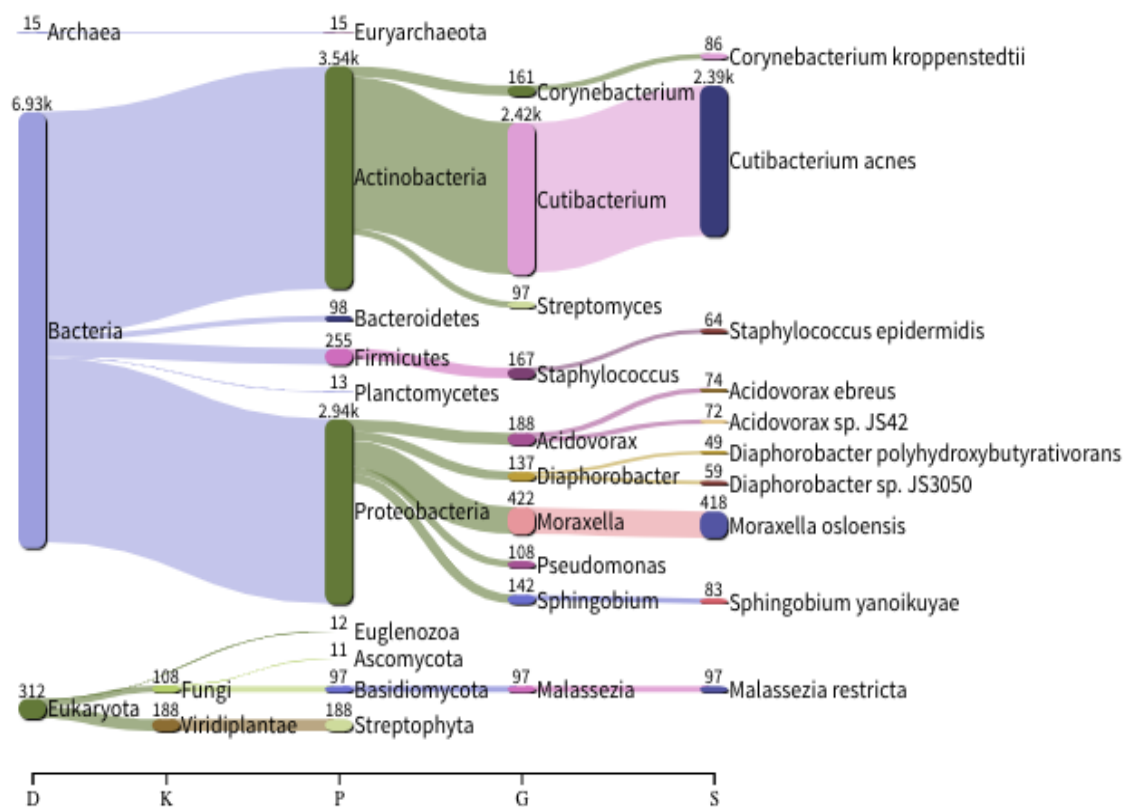
```
kraken2 -db standard_db -threshold 0.10 -threads 8 -use-names -fastq-input -report out.report -gzip-compressed in_1.fastq.gz in_2.fastq.gz
```

See the kraken2 manual for more information: <https://github.com/DerrickWood/kraken2/wiki/Manual>

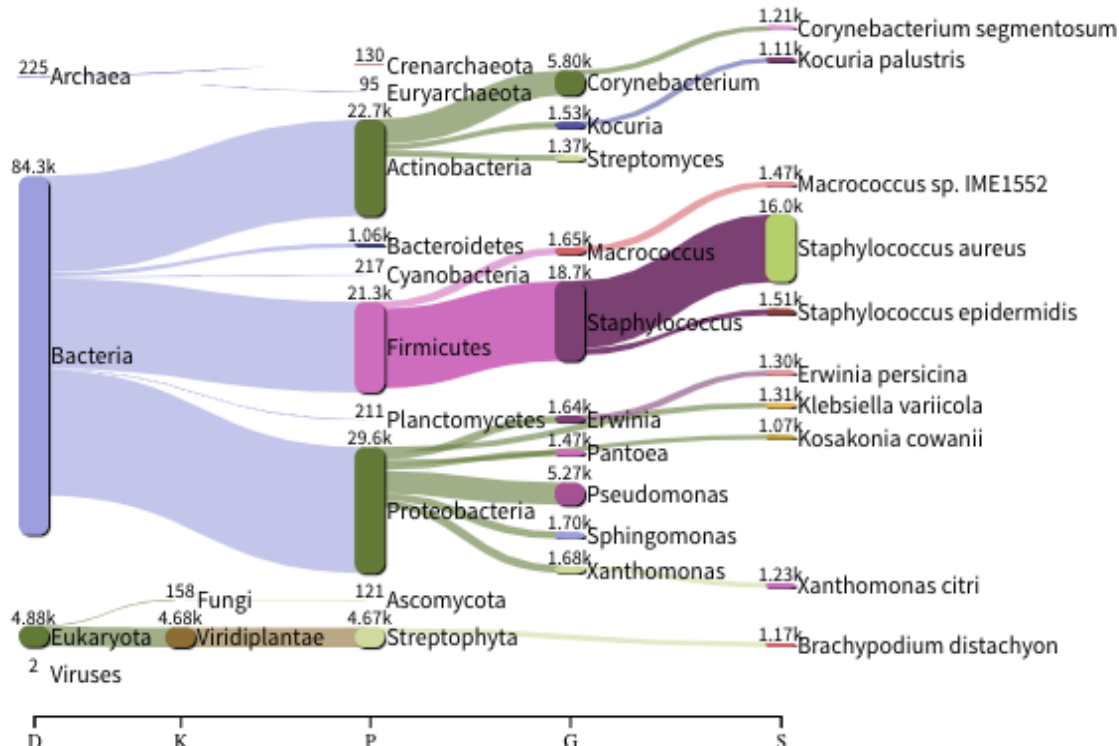
See Pavian manual for the plots: <https://ccb.jhu.edu/software/pavian/>



The following image shows the microbial abundance in the negative control



The following image shows the microbial abundance in the skin sample



Look for similarities and differences at both the phylum and genus level - labelled as 'P' and 'G' on the bottom axis.



Is there any overlap between the negative control and skin sample phylum? Can we map the negative control directly to the skin sample to remove all contaminants? If not, why?



Are there any genera in the negative control which aren't present in the skin sample? If you do a google search of this genus, where are they commonly found? With this information, where could this bacteria in the negative control have originated from?



For more practice assessing and trimming datasets, there is another set of raw reads called "skin\_example\_aa" from the skin metagenome available. These will require a fastqc or multiqc report, followed by trimming and mapping to the reference database with kneaddata. Using what you have learned previously, construct the relevant commands. Remember to check the quality before and after trimming.

Hint: Consider other trimmomatic options from the manual [http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/TrimmomaticManual\\_V0.32.pdf](http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/TrimmomaticManual_V0.32.pdf) e.g. "ILLUMINACLIP", where /opt/data/NexteraPE-PE is a file of adapters.



Navigate to skin folder and run quality control

```
cd /opt/data/skin
<construct the required commands>
```

## 1.5 MGnify API data hunt

This tutorial provides an introduction to the API (Application Programming Interface) and tools & methods that can be used to access microbiome data programmatically from MGnify. You will learn about the structure of the API and the data, as well as how to write scripts to analyze data programmatically.



These instructions assume you are using the EBI Training Virtual Machines. If you're running this on your own computer instead, remember that some file paths will be different. In particular, `/home/training/` will be `/home/<yourusername>`.

### 1.5.1 Learning objectives

- Understand how to access data using MGnify API
- Understand how to filter data sets using metadata
- Learn how to write scripts to programmatically access to the data

### 1.5.2 Prerequisites

For this tutorial you will need to install the software using [miniconda](#) and a working directory to store the data.

Miniconda has been pre-installed in the VM but it is easy to install, [instructions here](#).

Open the terminal and execute the following instructions:

```
mkdir -p /home/training/api_session

cd /home/training/api_session

conda create -n mgnify-api python=3.8

conda activate mgnify-api

pip install pandas numpy scipy plotnine jsonapi-client mg-toolkit requests

curl http://ftp.ebi.ac.uk/pub/databases/metagenomics/mgnify_courses/ebi_2020/api.tar.gz_
↪ | tar xz --strip 1
```

You have just created a directory to store all the data, installed the required software and downloaded the scripts and expected results from the EBI ftp server.

### 1.5.3 An introduction to MGnify REST API

MGnify is a freely available hub for the analysis and exploration of metagenomic, metatranscriptomic, amplicon and assembled datasets. The resource provides rich functional and taxonomic analyses of user-submitted sequences, as well as analysis of publicly available metagenomic datasets drawn from the European Nucleotide Archive (ENA).

## How to browse data using MGnify REST API

The MGnify REST API allows retrieval of over 400.000 (and counting) publicly available metagenomics, metatranscriptomic, amplicon and assembly datasets, sampled from diverse environments.

The base URL to the API is: <https://www.ebi.ac.uk/metagenomics/api>

The API documentation at: <https://www.ebi.ac.uk/metagenomics/api/docs>

The base URL provides access to several resource collections, such as *studies* **samples**, **runs**, **analyses**, **genomes**, **biomes** and **experiment-types**



Open <https://www.ebi.ac.uk/metagenomics/api/latest/studies> in your browser. This will a paginated list of all of the publicly available studies. Now open the list of samples using: <https://www.ebi.ac.uk/metagenomics/api/latest/samples>



Details about a single project can be retrieved by providing a unique identifier assigned during the archiving process. For example, <https://www.ebi.ac.uk/metagenomics/api/latest/studies/ERP009703> provides access to the Ocean Sampling Day (OSD) 2014 project.



Retrieve the list of samples contained in this study using the following URL: <https://www.ebi.ac.uk/metagenomics/api/latest/studies/ERP009703/samples>. Explore the response, at the bottom of the page you can find the number of pages that match this query.



Now, retrieve all the analyses performed on this study using: <https://www.ebi.ac.uk/metagenomics/api/latest/studies/ERP009703/analyses>.



Question 1: Is the number of samples the same as the number of analyses?. What could be the reason?



Parameters can be added to the URL to filter and sort the data, allowing the construction of more complex queries. The API browser lists the filters that are available, as illustrated in Figures 2 and 3.



Question 2: Using the API browser, how many results have been analysed with the pipeline version 4.0 for the OSD study ERP009703?

### 1.5.4 Programmatic access

In the next few exercises we are going to utilize some Python scripts to interact with the MGnify REST API programmatically.

The data and scripts are also available in the [source code of this documentation](#).

#### Data exchange format

The industry default data exchange format for Web API is JSON. This format is a compact and human-readable way of representing data. A brief overview of the format [json](#).

The MGnify REST API returns a JSON object formatted data structure that contains the resource type, associated object identifier, attributes and relationships to other resources, allowing the construction of complex queries.

Standardized format data structures allow third party libraries in many programming languages to easily access data programmatically.

EMBL-EBI

Services

Research

Training

About us

Q

EMBL-EBI

Hinxton

# MGNify

Submit, analyse, discover and compare microbiome data

OverviewAPIAPI DocumentationAboutHelp

HOME / API BROWSER

Login

OPTIONS

GET


## Api Browser

MGNify API provides programmatic access to the data for cross-database complex queries. For more details review the documentation.

```
GET /metagenomics/api/v1/

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/vnd.api+json
Vary: Accept

{
  "data": {
    "biomes": "https://www.ebi.ac.uk/metagenomics/api/v1/biomes",
    "studies": "https://www.ebi.ac.uk/metagenomics/api/v1/studies",
    "super-studies": "https://www.ebi.ac.uk/metagenomics/api/v1/super-studies",
    "samples": "https://www.ebi.ac.uk/metagenomics/api/v1/samples",
    "runs": "https://www.ebi.ac.uk/metagenomics/api/v1/runs",
    "assemblies": "https://www.ebi.ac.uk/metagenomics/api/v1/assemblies",
    "analyses": "https://www.ebi.ac.uk/metagenomics/api/v1/analyses",
    "experiment-types": "https://www.ebi.ac.uk/metagenomics/api/v1/experiment-types",
    "pipelines": "https://www.ebi.ac.uk/metagenomics/api/v1/pipelines",
    "pipeline-tools": "https://www.ebi.ac.uk/metagenomics/api/v1/pipeline-tools",
    "publications": "https://www.ebi.ac.uk/metagenomics/api/v1/publications",
    "genomes": "https://www.ebi.ac.uk/metagenomics/api/v1/genomes",
    "release": "https://www.ebi.ac.uk/metagenomics/api/v1/release",
    "genomeset": "https://www.ebi.ac.uk/metagenomics/api/v1/genomeset",
    "cogs": "https://www.ebi.ac.uk/metagenomics/api/v1/cogs",
    "kegg-modules": "https://www.ebi.ac.uk/metagenomics/api/v1/kegg-modules",
    "kegg-classes": "https://www.ebi.ac.uk/metagenomics/api/v1/kegg-classes",
    "antismash-geneclusters": "https://www.ebi.ac.uk/metagenomics/api/v1/antismash-geneclusters",
    "annotations/go-terms": "https://www.ebi.ac.uk/metagenomics/api/v1/annotations/go-terms",
    "annotations/interpro-identifiers": "https://www.ebi.ac.uk/metagenomics/api/v1/annotations/interpro-identifiers",
    "annotations/kegg-modules": "https://www.ebi.ac.uk/metagenomics/api/v1/annotations/kegg-modules",
    "annotations/pfam-entries": "https://www.ebi.ac.uk/metagenomics/api/v1/annotations/pfam-entries",
    "annotations/kegg-orthologs": "https://www.ebi.ac.uk/metagenomics/api/v1/annotations/kegg-orthologs",
    "annotations/genome-properties": "https://www.ebi.ac.uk/metagenomics/api/v1/annotations/genome-properties",
    "annotations/antismash-gene-clusters": "https://www.ebi.ac.uk/metagenomics/api/v1/annotations/antismash-gene-clusters",
    "annotations/organisms": "https://www.ebi.ac.uk/metagenomics/api/v1/annotations/organisms",
    "mydata": "https://www.ebi.ac.uk/metagenomics/api/v1/mydata"
  }
}
```

 MGNify is part of the ELIXIR infrastructure  
[Learn more](#)

EMBL-EBI

Services

Research

Training

Industry

About EMBL-EBI

By topic

By name (A-Z)

Help & Support

Publications

Research groups

Postdocs & PhDs

Train at EBI

Train outside EBI

Train online

Contact organisers

Members Area

Workshops

SME Forum

Contact Industry programme

Contact us

Events

Jobs

News

People & groups

EMBL-EBI, Wellcome Genome Campus, Hinxton, Cambridgeshire, CB10 1SD, UK. +44 (0)1223 49 44 44  
Copyright © EMBL 2020 | EMBL-EBI is part of the European Molecular Biology Laboratory | [Terms of use](#)

Intranet

Fig. 1: **Figure 1:** MGNify API browser.

The screenshot displays the MGnify website header with the logo and navigation links. Below the header, the 'Genome List' section is visible. A red box highlights the 'Filters' button in the top right corner, with the text 'Filters menu' written below it. Below the filters menu, a pagination bar shows page numbers 1, 2, 3, ..., 186. The main content area displays the raw JSON response for the GET /metagenomics/api/v1/genomes endpoint.

**MGnify**  
Submit, analyse, discover and compare microbiome data

Overview | **API** | API Documentation | About | Help | Login

HOME / API BROWSER / GENOME LIST

**Genome List**

**Filters menu**

« 1 2 3 ... 186 »

```
GET /metagenomics/api/v1/genomes

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/vnd.api+json
Vary: Accept

{
  "links": {
    "first": "https://www.ebi.ac.uk/metagenomics/api/v1/genomes?page=1",
    "last": "https://www.ebi.ac.uk/metagenomics/api/v1/genomes?page=186",
    "next": "https://www.ebi.ac.uk/metagenomics/api/v1/genomes?page=2",
    "prev": null
  },
  "data": [
    {
      "type": "genomes",
      "id": "MGYG-HGUT-04644",
      "attributes": {
        "genome-id": 3094,
        "geographic-origin": "Europe",
        "geographic-range": [
          "North America",

```

Fig. 2: **Figure 2:** Filters menu in MGnify API browser.

Filters

### Field filters

Accession:

Select by MGnify, NCBI, IMG or Patric accession

Taxon lineage:

Taxon lineage

Type:

 MAG

or isolate

Length is greater than or equal to:

Length (bp) greater/equal value

Length is less than or equal to:

Length (bp) less/equal value

Number of contigs:

Fig. 3: **Figure 3:** Filters pop up menu for the Genomes list endpoint.



```

GET /metagenomics/api/v1/studies

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/vnd.api+json
Vary: Accept

{
  "links": {
    "first": "https://www.ebi.ac.uk/metagenomics/api/v1/studies?page=1",
    "last": "https://www.ebi.ac.uk/metagenomics/api/v1/studies?page=163",
    "next": "https://www.ebi.ac.uk/metagenomics/api/v1/studies?page=2",
    "prev": null
  },
  "data": {
    {
      "type": "studies",
      "id": "MGYS00005605",
      "attributes": {
        "samples-count": 228,
        "accession": "MGYS00005605",
        "bioproject": "PRJEB39913",
        "secondary-accession": "ERP123465",
        "centre-name": "VETERINARY Faculty",
        "is-public": true,
        "public-release-date": null,
        "study-abstract": "It is well established that the gastrointestinal (GI) microbiota regulate immune responses to GI pathogens. In particular, data from",
        "study-name": "The impact of helminth infection on the mucosal and luminal GI microbiota of the horse",
        "data-origination": "SUBMITTED",
        "last-update": "2020-10-30T08:59:41"
      },
      "relationships": {
        "publications": {
          "links": {
            "related": "https://www.ebi.ac.uk/metagenomics/api/v1/studies/MGYS00005605/publications"
          }
        },
        "downloads": {
          "links": {
            "related": "https://www.ebi.ac.uk/metagenomics/api/v1/studies/MGYS00005605/downloads"
          }
        },
        "geocoordinates": {
          "links": {
            "related": "https://www.ebi.ac.uk/metagenomics/api/v1/studies/MGYS00005605/geocoordinates"
          }
        }
      }
    }
  }
}

```

Fig. 4: **Figure 4:** MGnify response output in JSON format.

## Exercise 1

In this exercise you will browse sample metadata and visualise analysis results. First we are going to look at retrieving samples that match particular metadata search criteria.



Read the code of the `exercise1.py` script. This script is using the API to obtain a subset of the samples.



Question 3: What “type” of data is the script downloading?. Which filters are being used to get the filtered data from the API?.



Run the script `exercise1.py` in the console:

```
python exercise1.py
```



Using these few lines of Python, we are able to retrieve the complete set of oceanographic samples taken at the Arctic Ocean (latitude > 70) across all publicly available studies in MGnify.



Inspect the generated `exercise1.csv` file.



Question 4: How might you adapt the script to find soil samples taken at the equator?.



Add an additional parameter “study\_accession”: “MGYS00000462” to the filters section in the script `exercise1.py` and run it again. You can check the study in the website [MGYS00000462](https://www.ebi.ac.uk/metagenomics/api/v1/studies/MGYS00000462).



Question 5: How many of the OSD2014 samples were taken from the Arctic Ocean?

	A	B	C	D	E	F	G
1	accession	sample-name	longitude	latitude	geo-loc-name	study	biome
2	ERS089005	Arctic seawater sample collected on 10 March 2008	-123.9109	71.0386	Arctic Ocean	MGYS00000297	root:Environmental:Aquatic:Marine
3	ERS1451501	North Sea - 1N	11.9099	78.9235	North Sea	MGYS00001316	root:Environmental:Aquatic:Marine
4	ERS1568962	KD006_S11_87	-176.7614	88.4072	Arctic Ocean	MGYS00000991	root:Environmental:Aquatic:Marine:Oceanic
5	ERS1568962	KD006_S11_87	-176.7614	88.4072	Arctic Ocean	MGYS00002040	root:Environmental:Aquatic:Marine:Oceanic
6	ERS1568963	KD008_S1_4	-176.7614	88.4072	Arctic Ocean	MGYS00000991	root:Environmental:Aquatic:Marine:Oceanic
7	ERS1568963	KD008_S1_4	-176.7614	88.4072	Arctic Ocean	MGYS00002040	root:Environmental:Aquatic:Marine:Oceanic
8	ERS1568964	KD010_S2_6	-176.7614	88.4072	Arctic Ocean	MGYS00000991	root:Environmental:Aquatic:Marine:Oceanic
9	ERS1568964	KD010_S2_6	-176.7614	88.4072	Arctic Ocean	MGYS00002040	root:Environmental:Aquatic:Marine:Oceanic
10	ERS1568965	KD017_S12_85B	-89.2525	89.9903	Arctic Ocean	MGYS00000991	root:Environmental:Aquatic:Marine:Oceanic
11	ERS1568965	KD017_S12_85B	-89.2525	89.9903	Arctic Ocean	MGYS00002040	root:Environmental:Aquatic:Marine:Oceanic
12	ERS1568966	KD028_S10_83	-148.8943	87.772	Arctic Ocean	MGYS00000991	root:Environmental:Aquatic:Marine:Oceanic
13	ERS1568966	KD028_S10_83	-148.8943	87.772	Arctic Ocean	MGYS00002040	root:Environmental:Aquatic:Marine:Oceanic
14	ERS1568967	KD029_S10_82	-148.8943	87.772	Arctic Ocean	MGYS00000991	root:Environmental:Aquatic:Marine:Oceanic
15	ERS1568967	KD029_S10_82	-148.8943	87.772	Arctic Ocean	MGYS00002040	root:Environmental:Aquatic:Marine:Oceanic
16	ERS1568968	KD030_S9_81	-148.8943	87.772	Arctic Ocean	MGYS00000991	root:Environmental:Aquatic:Marine:Oceanic
17	ERS1568968	KD030_S9_81	-148.8943	87.772	Arctic Ocean	MGYS00002040	root:Environmental:Aquatic:Marine:Oceanic
18	ERS1568969	KD031_S8_80	-148.8943	87.772	Arctic Ocean	MGYS00000991	root:Environmental:Aquatic:Marine:Oceanic
19	ERS1568969	KD031_S8_80	-148.8943	87.772	Arctic Ocean	MGYS00002040	root:Environmental:Aquatic:Marine:Oceanic
20	ERS1568970	KD032_S7_79	-148.8943	87.772	Arctic Ocean	MGYS00000991	root:Environmental:Aquatic:Marine:Oceanic
21	ERS1568970	KD032_S7_79	-148.8943	87.772	Arctic Ocean	MGYS00002040	root:Environmental:Aquatic:Marine:Oceanic
22	ERS1568971	Arctic sea ice metagenome, 1.5 m	-148.8943	87.772	Arctic Ocean	MGYS00000991	root:Environmental:Aquatic:Marine:Oceanic
23	ERS1568971	Arctic sea ice metagenome, 1.5 m	-148.8943	87.772	Arctic Ocean	MGYS00002040	root:Environmental:Aquatic:Marine:Oceanic

Fig. 5: **Figure 5:** Exercise 1 retrieved data in CSV format.

## Exercise 2

For this exercise we will use the MGnify REST API to obtain data and then visualize the analysis results of the study “Metabolically active microbial communities in marine sediment under high-CO<sub>2</sub> and low-pH extremes MGYS00002474 (DRP001073). In this study, DNA was extracted from sub-seafloor sediments and domain specific 16S rRNA gene primers were used to profile the archaeal and bacterial taxonomic communities.

We will begin by retrieving taxonomic analysis data and then plotting relative abundance in the form of bar charts.



Open the file `exercise2.py`. Read the code, even if you don't understand python the variables and constants at the beginning of the file will allow the script to be easily modified. Note, there are a series study accessions that are currently commented out which will allow you to rerun the analysis with other projects. Ignore them for the time being and run the code with MGYS00002474 and inspect the resultant bar chart.

```
python exercise2.py
```



Question 6: How similar or different are the phylum compositions of each analysis?. How might you explain any differences?.



Question 7: How many of the analyses look to target bacterial populations and how many are targeting the archaea?



It's easy to adapt the script for other analyses. For example, if you change the variable TAX\_RANK (line 13 in `exercise2.py`) to “genus” you can obtain the genus level results.



Question 8: How might you adapt the code for the analysis of other studies:

- perform analysis of taxonomic results bases on the large ribosomal subunit rRNA or the ITS region for fungi?
- output the top 20 genera, rather than the top 10?

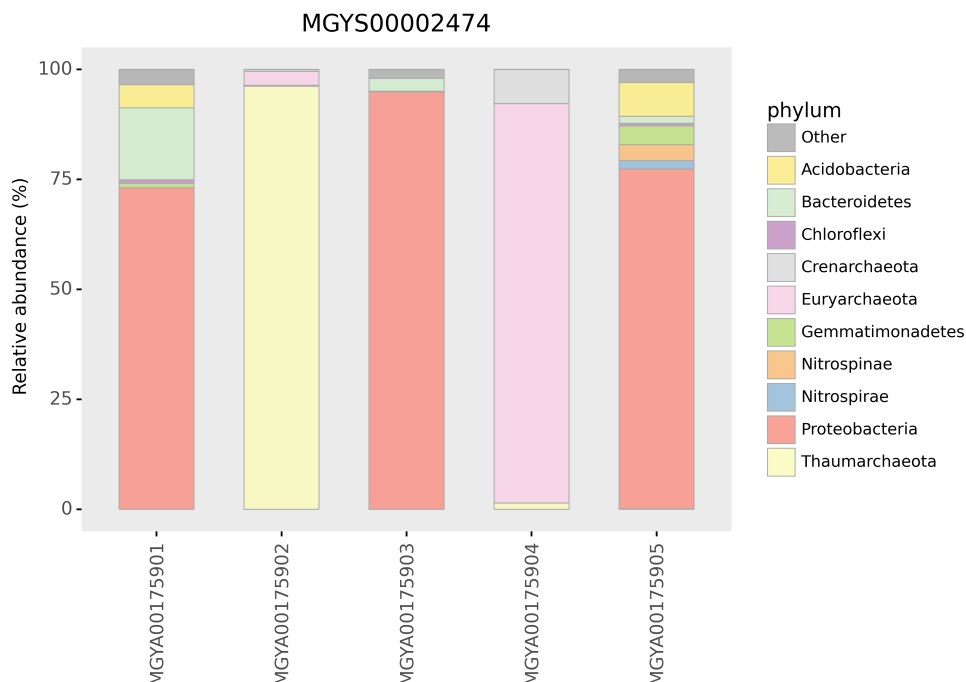


Fig. 6: **Figure 6:** Microbiome diversity in the study MGYS00002474 (DRP001073) at the phylum level.

### Exercise 3

For this exercise we will use the MGnify REST API and the [MG-Toolkit](#) to download analysis files. The toolkit is a command line application. At the moment of writing this the toolkit doesn't support MGnify genomes, for this resource we will use the MGnify REST API.



We installed the toolkit when we created the conda environment.



Question 9: What type of data can we download using the Toolkit?.



Download the metadata for the study "DRP001073 (MGYS00002474)". Execute the following command:

```
mg-toolkit original_metadata -a DRP001073
```



Explore the generated csv file DRP001073.csv. Can you notice if there is something wrong with the metadata?



Let's download the functional annotations for one study. Run the following command in the terminal:

```
mg-toolkit bulk_download -a MGYS00005575 --result_group pathways_and_systems
```



This may take a few minutes.



Question 10: Based on the files the toolkit has downloaded, how many analyses has the study MGYS00005575?.



For the genome resource we will use the API directly to obtain all the Download files for the genome: MGYG-

HGUT-04644



Open the file `exercise3.py`. Read the code. This script will download all the files from the download's relationship for the selected genome.

Execute the script to fetch the files:

```
python exercise3.py
```



Explore the different files that were downloaded.



Question 11: How could you modify the script *exercise3.py* to download the other functional annotations?.

### 1.5.5 Solutions

To get the solutions to all the questions and R versions of the scripts [follow this link](#).

## 1.6 License

This work is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.