Course Project 2:
Microbiome data analysis of Zygnema green algae

Xinpeng Zhang (PhD student)
Dr. Xuehuan Feng(Postdoc)
Department of Food Science and Technology

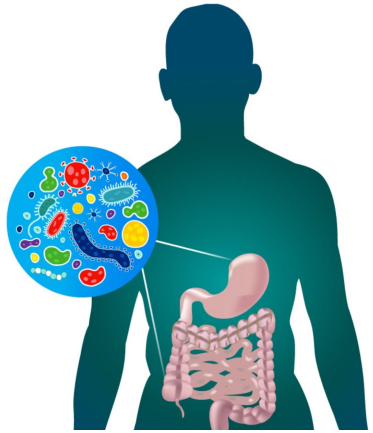
11/21/2023

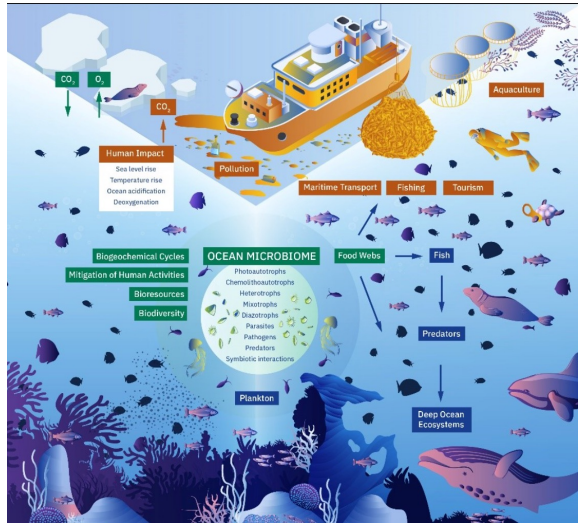
Dr. Yin's Lab at UNL

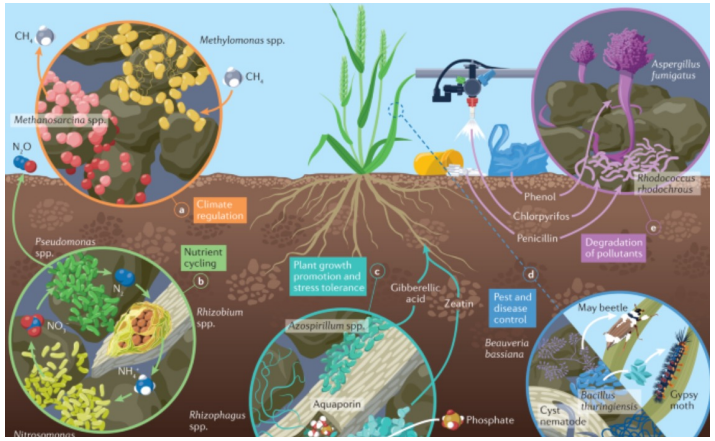
PI: Yanbin Yin

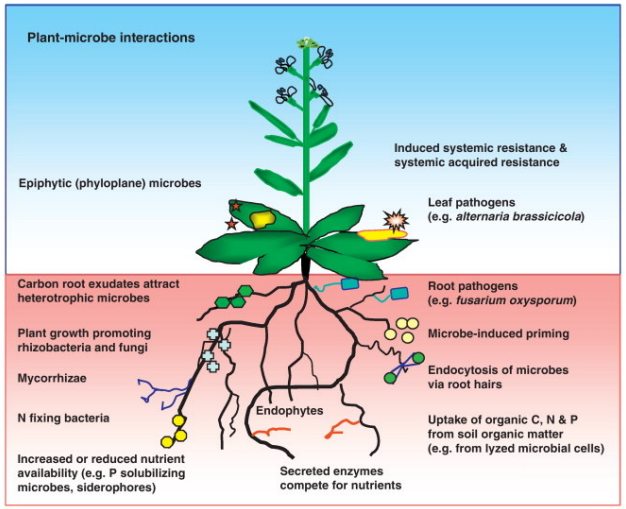
What is microbiome

- Microbiota refers to all microorganisms (bacteria, fungi, virus, etc.) within a given environment.
- Microbiome refers to the collection of genomes from microbiota in the environment.
- Examples of microbiome environment:
  - Human gut
  - Marine
  - Soil
  - Plant

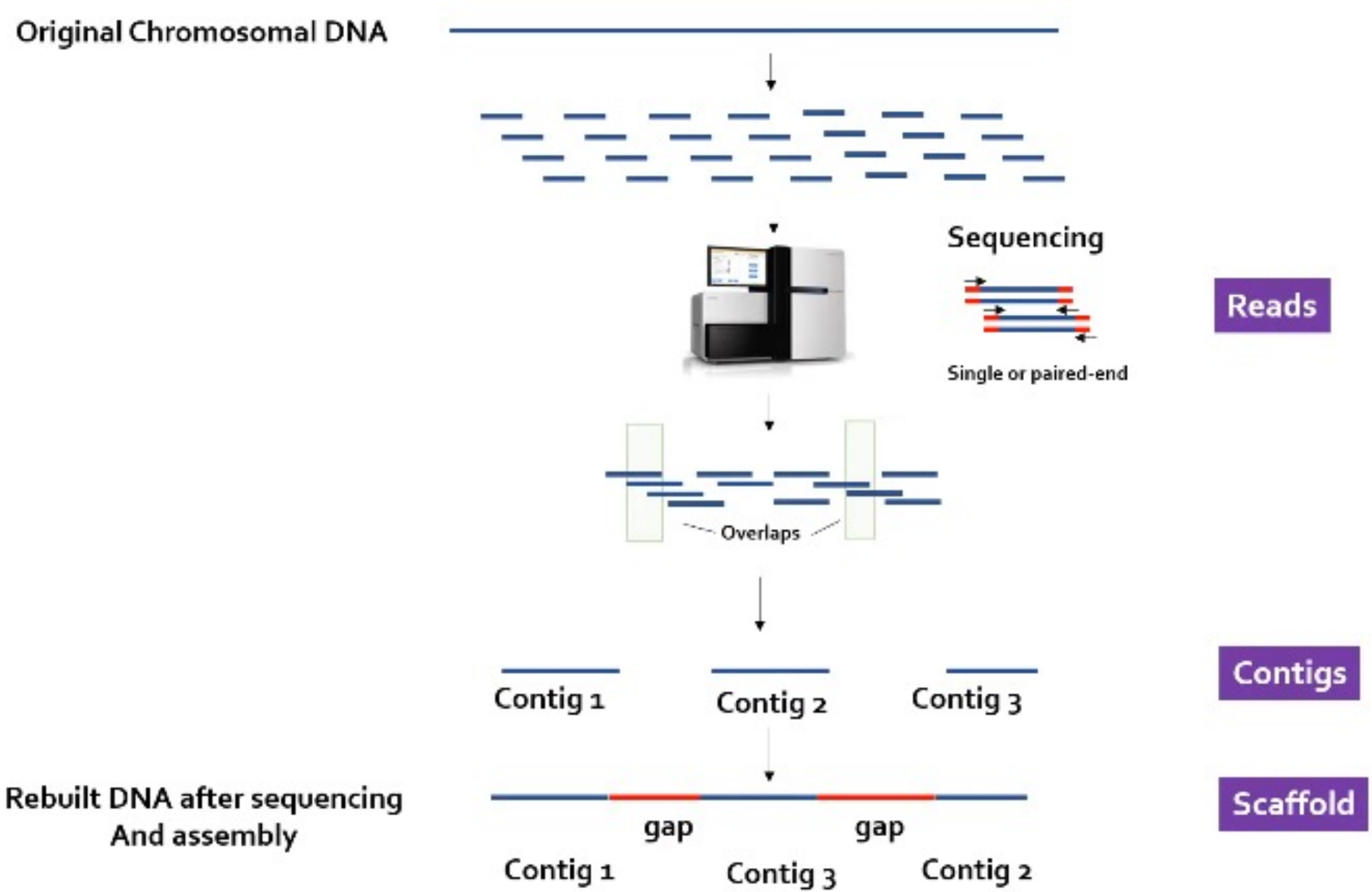








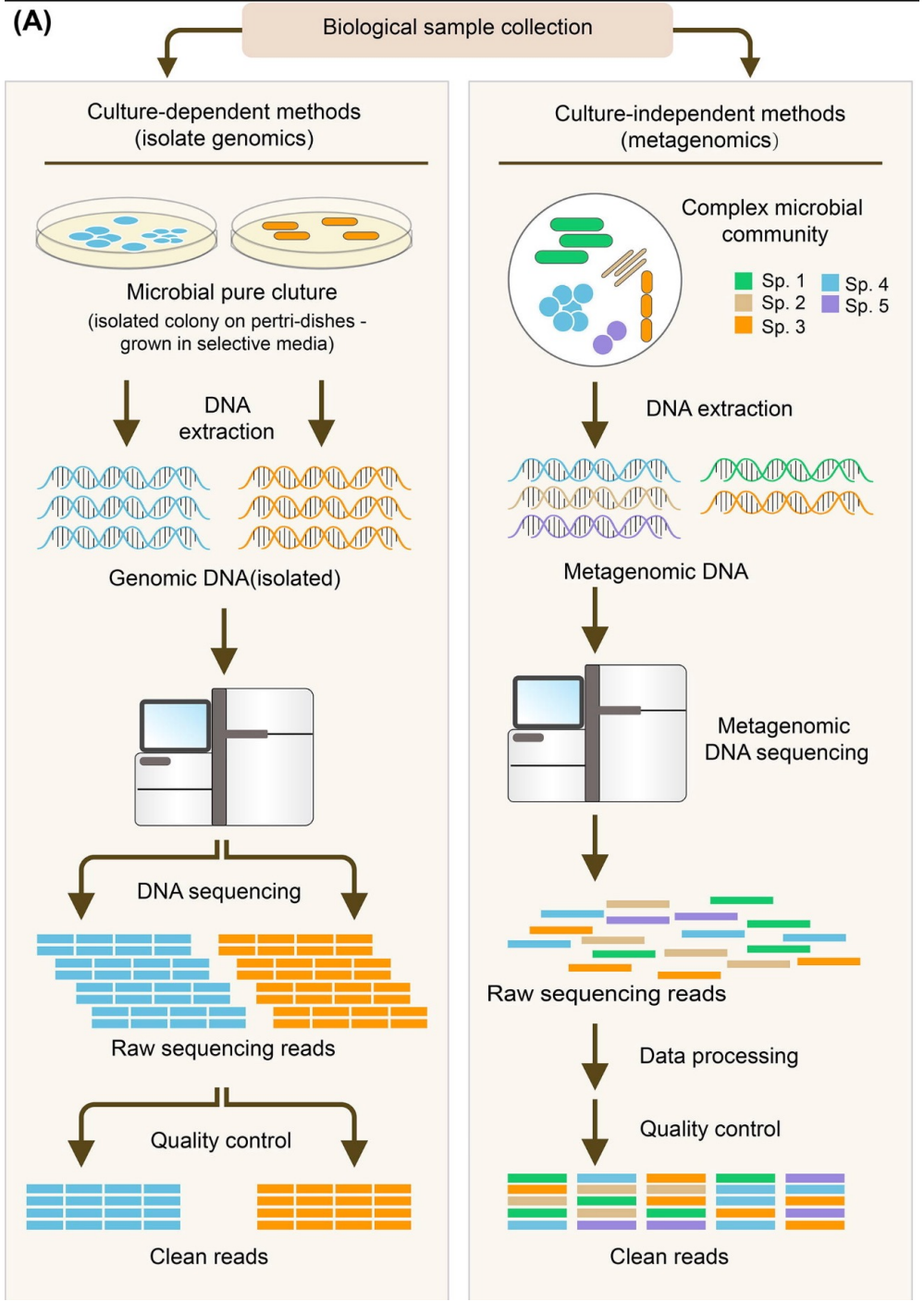
Pictures from google



https://www.youtube.com/watch?v=XPu_O6GIJI0

- One of the big questions in studying the microbiome is:
- Many microorganisms cannot be cultured for the time being.
- To fix this problem, metagenomics are developed:
- Metagenomics is the study of the structure and function of entire nucleotide sequences isolated and analyzed from all the organisms (typically microbes) in a bulk sample.

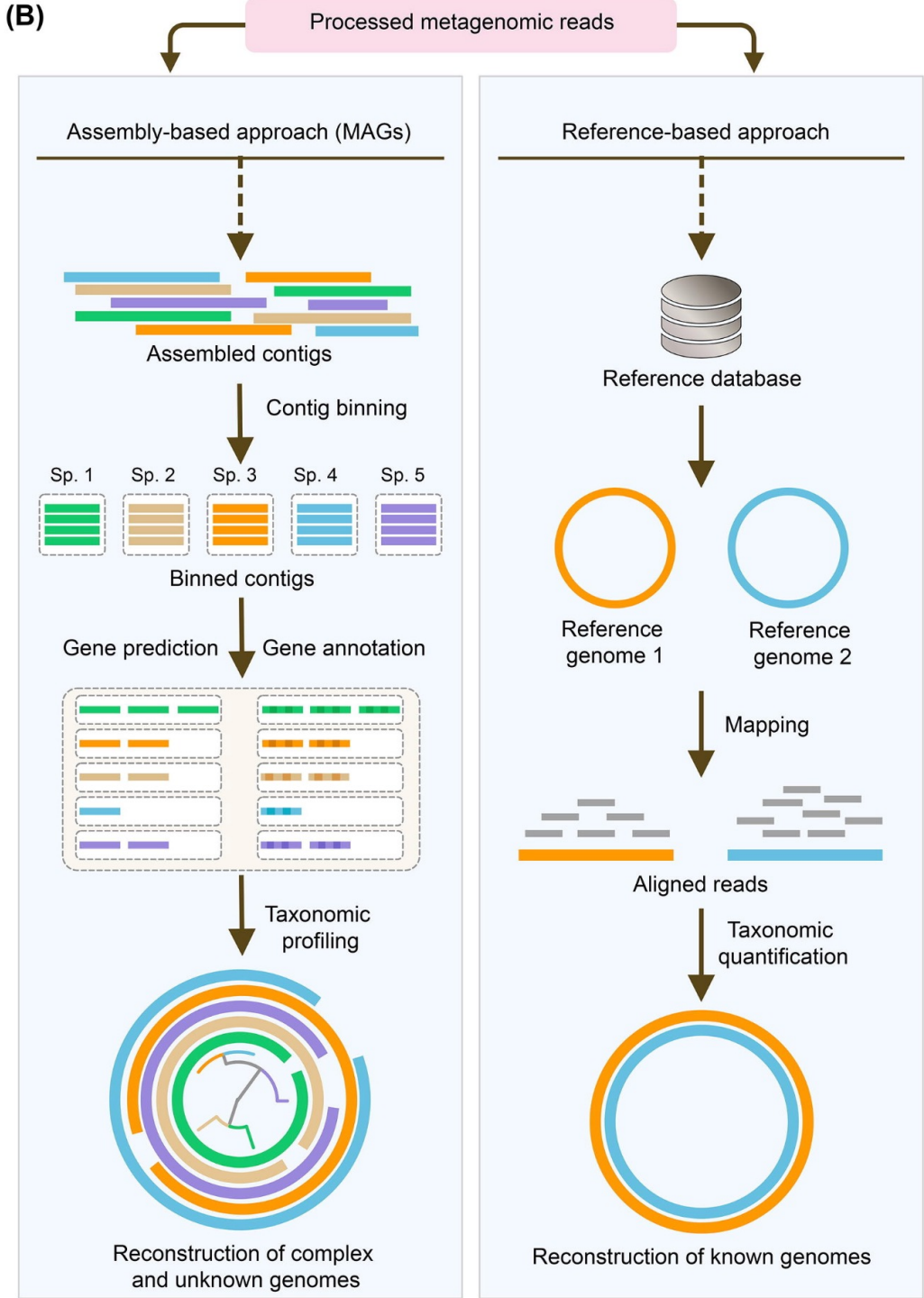
https://www.sciencedirect.com/science/article/pii/S2001037021004931



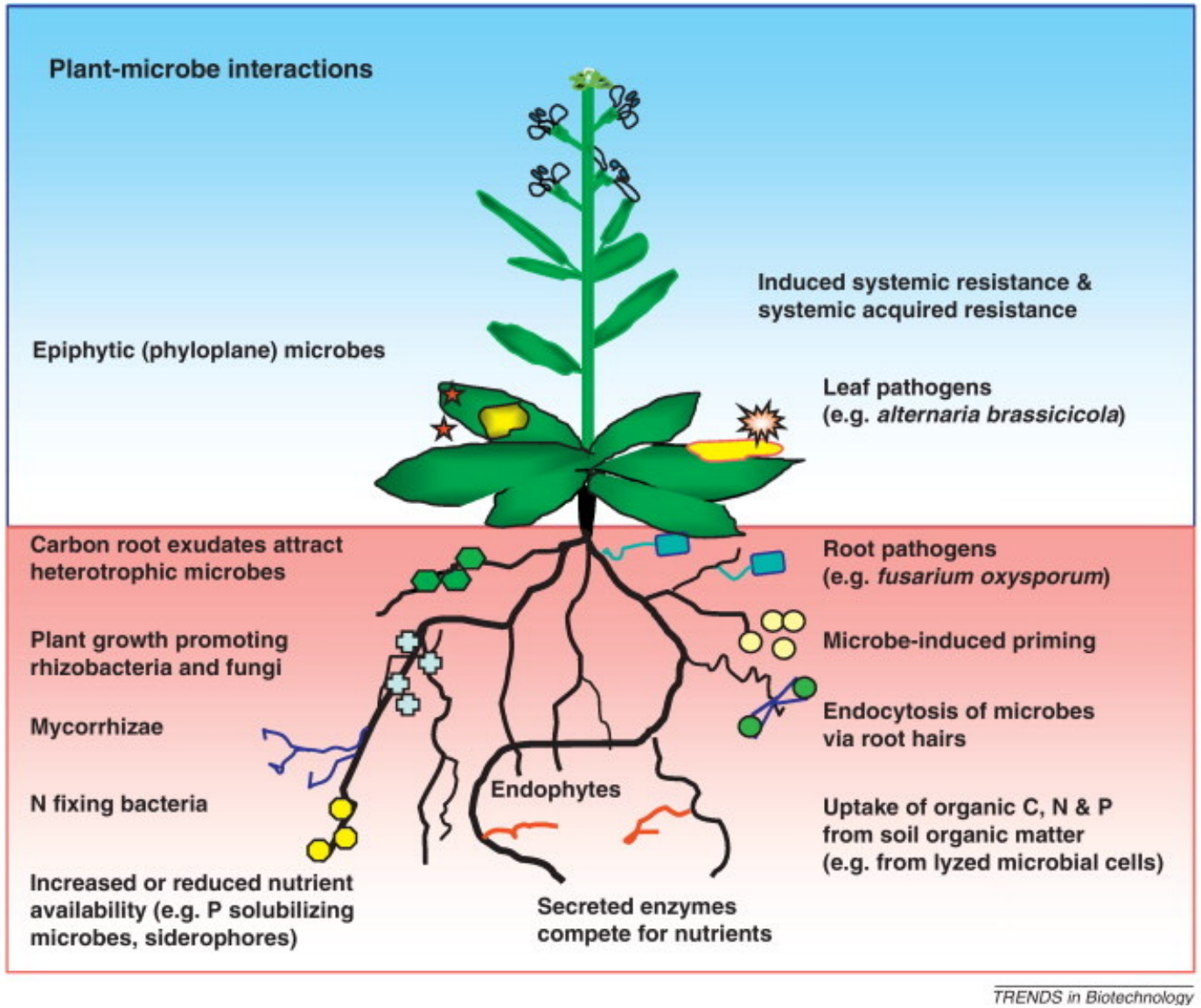
- On the left, raw reads are assembled by assembling them to form short contigs, which are then merged into larger bins based on different methods, and each bin is a genome, called Metagenome-Assembled Genomes (MAGs).

- On the right is a traditional method, by mapping raw reads with reference genomes (golden standards) obtained by other methods, and eventually reconstructing them into genomes based on the reference genome.

https://www.sciencedirect.com/science/article/pii/S2001037021004931



Plant-microbe interactions



- Plants in their **natural habitats** are surrounded by a large number of microorganisms.
- Some microbes directly interact with plants in a **mutually beneficial** manner whereas others colonize the plant only for their own benefit.
- Various plant–microbe interactions can be broadly categorized as **beneficial**, **detrimental** or **neutral**.
- The microbes may utilize plant-derived organic compounds as substrates for **energy production**.
- **Beneficial microbes** promote plant growth and/or suppress plant diseases via a variety of mechanisms, which include improved nutrient acquisition, production of growth regulators, and biosynthesis of pathogen-inhibiting compounds.
- For example, the nodulation recognition system in legumes.

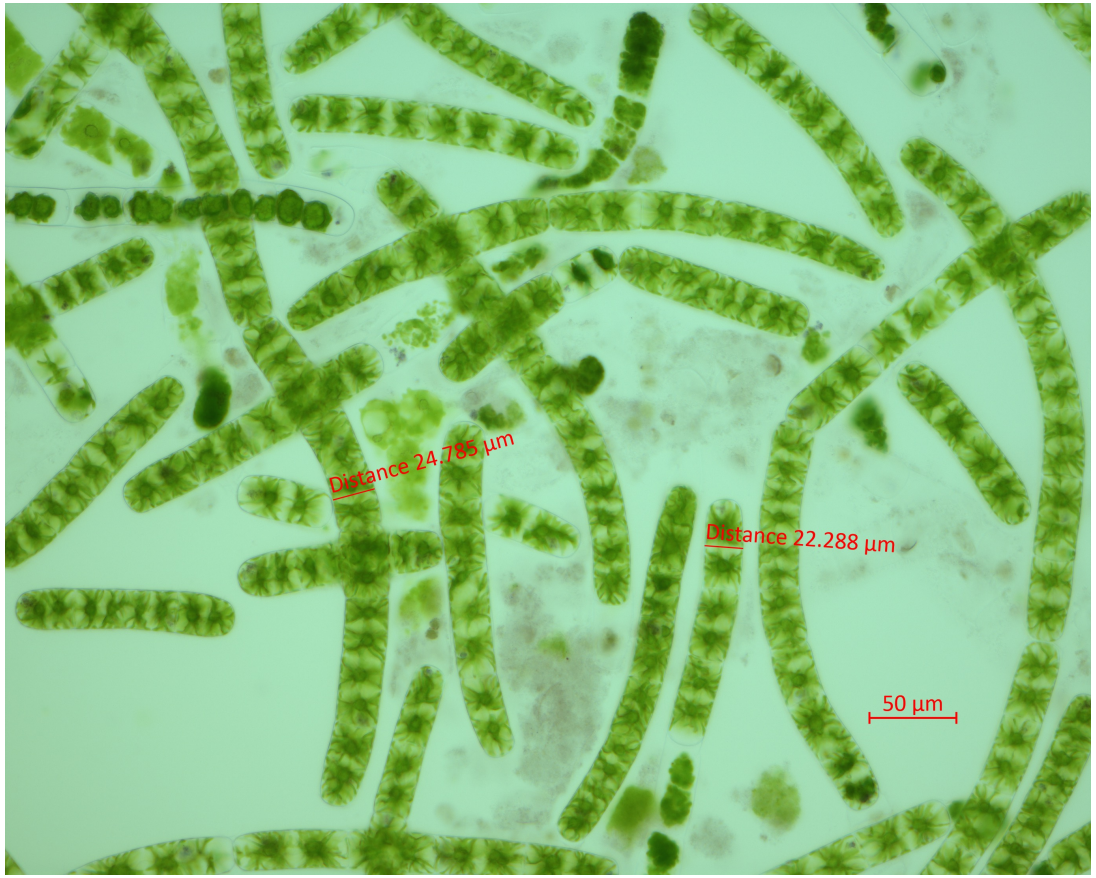
Schenk et al. (2012) Unraveling plant-microbe interactions: can multi-species transcriptomics help? Trends Biotechnol. 30(3):177-84.

- There are many reports on the interaction between land plants and microbes, but there are still much unknown about the relations between **algae and microbes**.

  For example:
  1. How the bacteria facilitate the algae adapt the land environment?
  2. Can bacteria help the algae against pathogens?
  3. Can bacteria induce the signaling transduction of the algae to response to environmental stress?
  4. Can bacteria produce chemicals to protect them from stress, such as cold stress? (directly or indirectly protect the algae.) What are the pathways involved in the responses?

Zygnema circumcarinatum MZCH241

MZCH-SVCK (University of Hamburg, Germany)
Microalgae and Zygnematophyceae Collection Hamburg



Zygnema circumcarinatum
MZCH241

https://www.biorxiv.org/content/10.1101/2023.01.31.526407v1

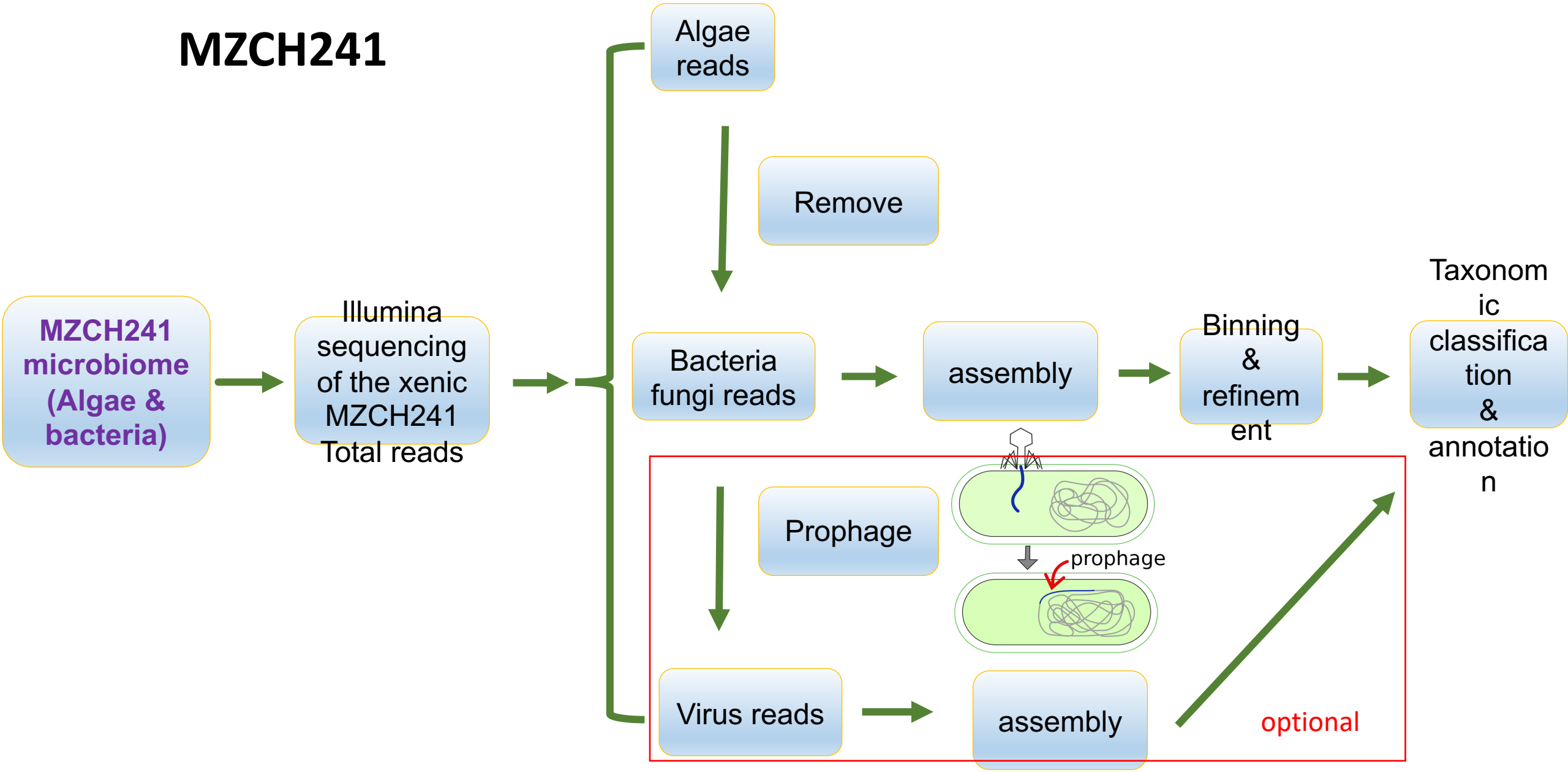
- In this project, we will learn how to do a complete metagenomic analysis, from raw reads to the gene annotations.

- We'll use algae microbiome as an example, and this protocol could be applied to other microbiome data.

- We'll provided you the metagenomic sequencing data(pair-end reads), including the algae genome and microbiome.

Microbiome workflow of *Zygnema circumcarinatum*

MZCH241



Recourse information

- Pre-installed programs on hcc: https://hcc.unl.edu/docs/applications/modules/

- Conda/mamba:

https://docs.conda.io/en/latest/
https://github.com/mamba-org/mamba
Conda/mamba is an python package and environment management system that runs on Windows, macOS, and Linux. Could create individual environments to avoid the program conflicts.
- Mamba is much faster than conda sometimes.

- A protocol command file has all details and commands.

Reminder before starting

1. For each step, create a folder with partN_name.
2. When we finish one part, remember to "module unload current_program" to avoid the program conflicts.
3. Do not forget where you are at, and the file path.
4. Do not run huge jobs directly on login node. Run it by submitting slurm scripts.
5. Do not forget to revise the slurm scripts to use your own path (err, out, input_file, etc.)
6. Read your error and output files for trouble shooting.
7. All commands and scripts are in a protocol command file.
8. Do it as soon as possible because you may need to wait for a longer time as HCC is busier at the end of the semester.

Sub-project I. Algal microbiome whole genome shotgun data analysis (required)

1. Metagenome sequencing
2. Metagenome-assembled genomes (MAGs)
3. MAGs binning & taxonomic classification

Part 0
Data preparation

- We will provide you the sequencing data.

- Create a folder for this part.

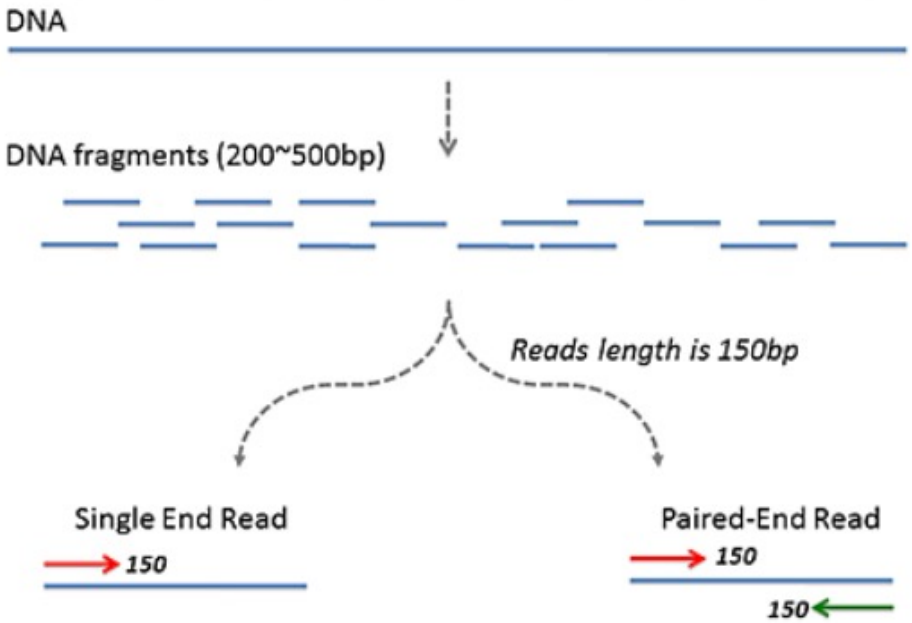
- Copy the data to your work folder.

```
-rwxr-xr-x 1 xinpeng yinlab 7935677320 Nov 13 15:10 M2CH_S1_L002_R1_001.fastq.gz
-rwxr-xr-x 1 xinpeng yinlab 8553228969 Nov 13 15:14 M2CH_S1_L002_R2_001.fastq.gz
```

```
#part0, raw data

#in this project, we'll work on metagenomic sequencing data of algae (Zygnematophyceae), which is the
paired-End sequencing. As a result, it contains two raw read files: fastq1 and fastq2. It includes algae
data and microbiomes data. We'll use different tools to remove those algae data and finally get microbiomes
data.
#we'll provide you the raw reads, you need to cp the folder by following steps:

cd $WORK
mkdir final_project
cd final_project
cp -r /work/yinlab/xinpeng/final_course_project/new/final_process/part0_raw_data .
```

Part I
Data preprocessing (~5 hrs)

- Next-generation sequencing(NGS) usually uses paired-ended sequencing, meaning that it is sequenced from two directions and ends up with two sequencing files, one forward and one reverse.



- We need to check the quality of sequencing data, such as low-quality reads and regions, contamination or technical deviations.

- By examining metrics such as sequence quality scores, sequence length distributions, GC content, repeat sequences, and k-mer content.

```
@A00419:304:HH53MDSXY:2:1101:2591:1000 1:N:0:CAACACAG+ATGTGAAG
GNTCTACCGATGTTCCTTATTATGTTGATAGCCCATTATCAGAGCAAGCTACACAGGTGTTAAAAGACCATCCGGAAGAATACAATA
AAAACGTAAAGAAAGTATTACAGGTAGATGCTGACCCATTCGGATTTAAAGGCTTAAAGTTTA
+
F#FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFF:FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
```

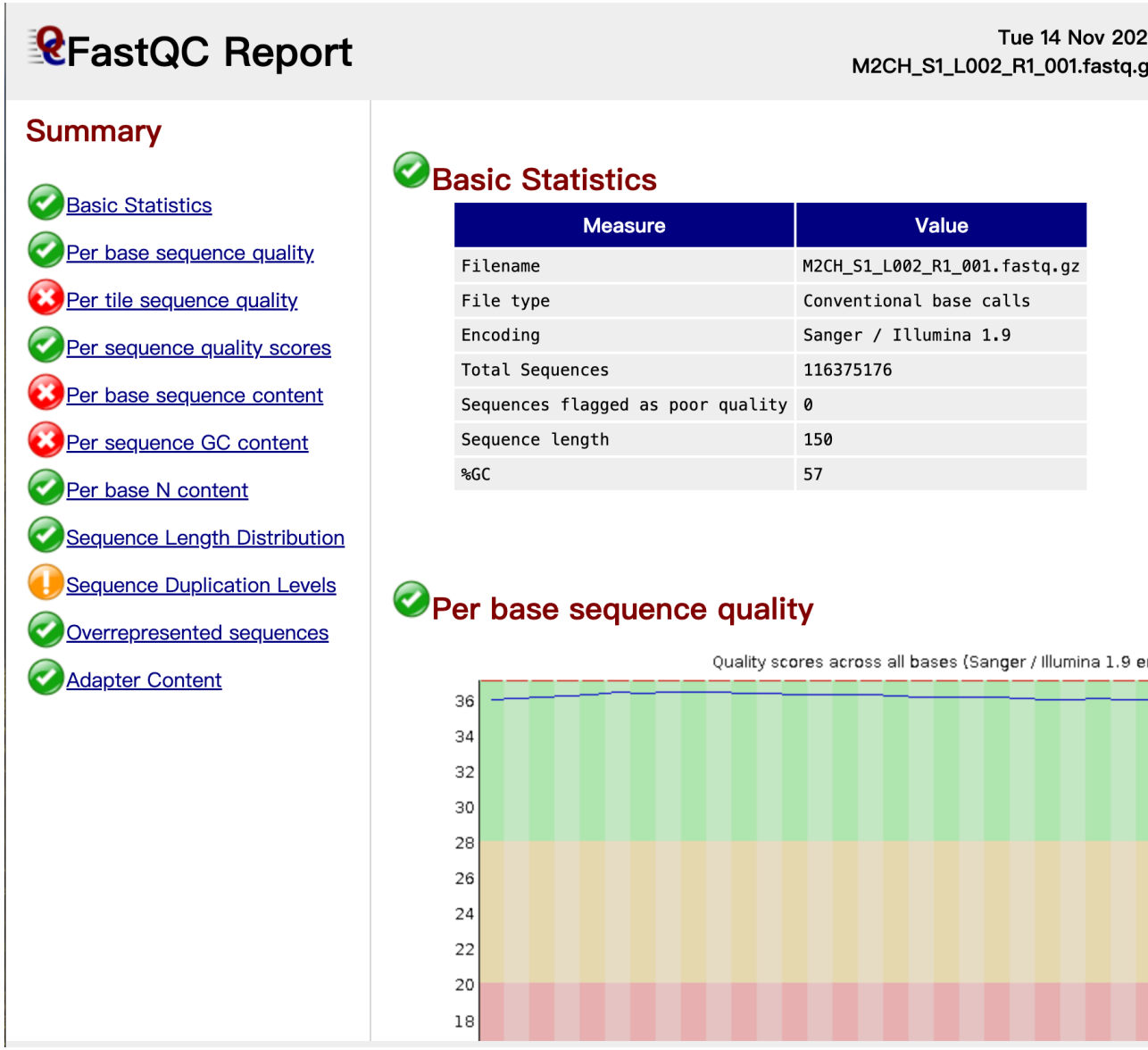
- First line: identifier and description (starts with @).
- Second line: sequence (nucleotide sequence).
- Third line: separator (starts with +).
- Fourth line: base quality scores (corresponds to the sequence in the second line).

```
@A00419:304:HH53MDSXY:2:1101:2591:1000 1:N:0:CAACACAG+ATGTGAAG
GNTCTACCGATGTTCCTTATTATGTTGATAGCCCATTATCAGAGCAAGCTACACAGGTGTTAAAAGACCATCCGGAAGAATACAATA
AAAACGTAAAGAAAGTATTACAGGTAGATGCTGACCCATTCGGATTTAAAGGCTTAAAGTTTA
+
F#FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFF:FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
```

- We'll use "FastQC" to check the quality of fastq file

```
mkdir part1_data_preprocessing
cd part1_data_preprocessing
ml fastqc
fastqc ../part0_raw_data/M2CH_S1_L001_R1_001.fastq.gz --outdir .
fastqc ../part0_raw_data/M2CH_S1_L002_R1_001.fastq.gz --outdir .
```

- After running fastqc, we will get a batch of results including a html file.
- Download and read it.
- In your report, you need to explain the meaning of each plot.



- After that, we would remove those adapters, low-quality reads and regions to avoid effecting our results, which is called "trim".
- We will use "trim_galore" which is preinstalled on hcc.
- https://github.com/FelixKrueger/TrimGalore
- There are different parameters, but here we just run with the default

```
#2. After that, we would delete those low quality reads by trimming using trim_galore (2-4hrs, depends the
cores you used).

#Remember to do module unload your_previous_program, because sometimes different program may cause
conflicts.
#won't remind you again.
#for the long time progress, we will offer you slurms to submit jobs on hcc, use nano to edit and save, and
then sbatch.

module unload fastqc


#!/bin/bash
#SBATCH --job-name=trim
#SBATCH --time=124:00:00
#SBATCH --mem=100gb
#SBATCH --partition=batch,guest (reminder: you could add your own partition, i.e. benson or vinlab)
#SBATCH --output=trim.%J.out
#SBATCH --error=trim.%J.err
#SBATCH --ntasks=20
#SBATCH --cpus-per-task=1

module load trim_galore/0.6
trim_galore --paired your/path/of/fastq1.gz_file  your/path/of/fastq2.gz_file  -j 20
```

-- Paired This parameter means the input files are paired-end sequencing

-j threads, many tools could use threads to increase the speed of computing by adding the amount of cores(CPUs/GPUs).

- After that, run the fastqc again to check the difference between the raw reads and trimmed ones.
- Also you could get the trimmed report by trim_galore.

```
[xinpeng@login1.swan trim]$ ll
total 15981380
-rw-r--r-- 1 xinpeng yinlab      13654 Nov 14 21:06 data.4397920.err
-rw-r--r-- 1 xinpeng yinlab          0 Nov 14 17:28 data.4397920.out
-rw-r--r-- 1 xinpeng yinlab       5108 Nov 14 18:38 M2CH_S1_L002_R1_001.fastq.gz_trimming_report.txt
-rw-r--r-- 1 xinpeng yinlab 7887135410 Nov 14 21:06 M2CH_S1_L002_R1_001_val_1.fq.gz
-rw-r--r-- 1 xinpeng yinlab       5364 Nov 14 21:06 M2CH_S1_L002_R2_001.fastq.gz_trimming_report.txt
-rw-r--r-- 1 xinpeng yinlab 8491702232 Nov 14 21:06 M2CH_S1_L002_R2_001_val_2.fq.gz
-rw-r--r-- 1 xinpeng yinlab        449 Nov 14 17:28 trim.sh
```

- Report the difference in your report.

Part II
Remove host reads (~ 1-2 hrs)

- The pair-end sequencing data contains not only microbiome data, but also host data: mitochondrial, chloroplast, and algal reads, etc.

- We need to remove those reads to avoid the contamination.

- We will use host reference genomes to remove those reads. The reference genomes (1a and 1b are two algae, and we have their nuclear, mitogenome, and plastid genomes) are sequenced by Yin lab (paper in review at Nature Genetics).

```
-rw-r--r-- 1 xinpeng yinlab    328791 Nov 14 20:50 1a_mitogenome_323370bp.fasta
-rw-r--r-- 1 xinpeng yinlab    167765 Nov 14 20:50 1a_plastid_165372bp.fasta
-rw-r--r-- 1 xinpeng yinlab    219818 Nov 14 20:50 1b_mitogenome_216190bp.fasta
-rw-r--r-- 1 xinpeng yinlab    160196 Nov 14 20:50 1b_platid_157548bp.fasta
-rw-r--r-- 1 xinpeng yinlab 364366604 Nov 14 20:50 SAG698_1a_genome.fna
-rw-r--r-- 1 xinpeng yinlab  72184238 Nov 14 20:50 SAG698_1b_genome.fna
```

- We will work on the trimmed fastq files.

- Combine all host reference genomes as one file first by "cat"

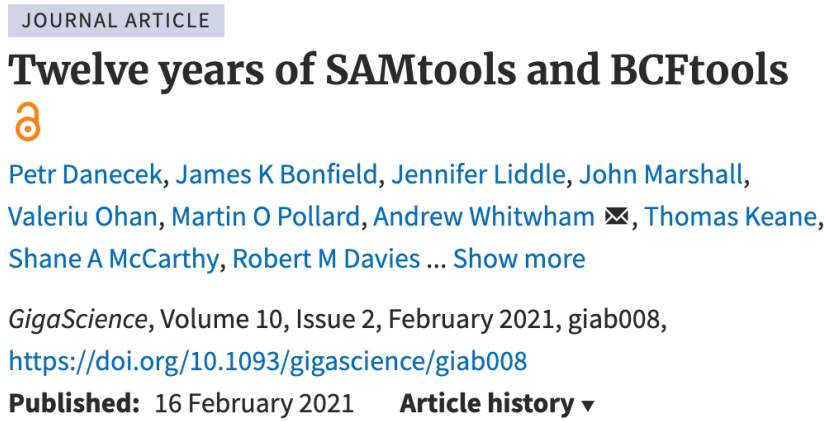
```
mkdir part2_remove_algea_genomes
cd part2_remove_algea_genomes
cp -r /work/yinlab/xinpeng/final_course_project/new/reference_genome .
cat * > algea_reference_genome.fna
```

- We will use "Bowtie2" and "SAMtools" to remove host reads from the sequencing data.
- Those are preinstalled on hcc.

- https://bowtie-bio.sourceforge.net/bowtie2/manual.shtml

- http://www.htslib.org/

- https://hcc.unl.edu/docs/applications/app_specific/bioinformatics_tools/alignment_tools/bowtie2/
- After running this step, we'll remove all host reads in both fastq files



```
#!/bin/bash
#SBATCH --job-name=Bowtie2
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=16
#SBATCH --time=168:00:00
#SBATCH --mem=30gb
#SBATCH --output=Bowtie2.%J.out
#SBATCH --error=Bowtie2.%J.err
#SBATCH --partition=yinlab,batch,guest

ml bowtie/2.3

ml samtools/1.3

bowtie2-build your_reference_genome_file index_prefix

bowtie2 -x index_prefix  -1 your/path/of/trimmed_fastq1.gz_file -2 your/path/of/
trimmed_fastq2.gz_file -S bowtie2_alignments.sam --local -p $SLURM_NTASKS_PER_NODE

samtools view -bS -@ 16 bowtie2_alignments.sam > bowtie2_alignments.bam

samtools sort -@ 16 bowtie2_alignments.bam -o bowtie2_alignments.sorted.bam

samtools index bowtie2_alignments.sorted.bam

samtools view -b -f 12 -F 256 -@ 16 bowtie2_alignments.sorted.bam > unmapped.bam

samtools fastq -@ 16 -1 unmapped_1.fastq -2 unmapped_2.fastq unmapped.bam
```

- SAM means “Sequence Alignment/Map format”. It is the result of alignment from fastq file.
- SAM file is really large.
- BAM is the compressed format of SAM, which is smaller and faster to view.

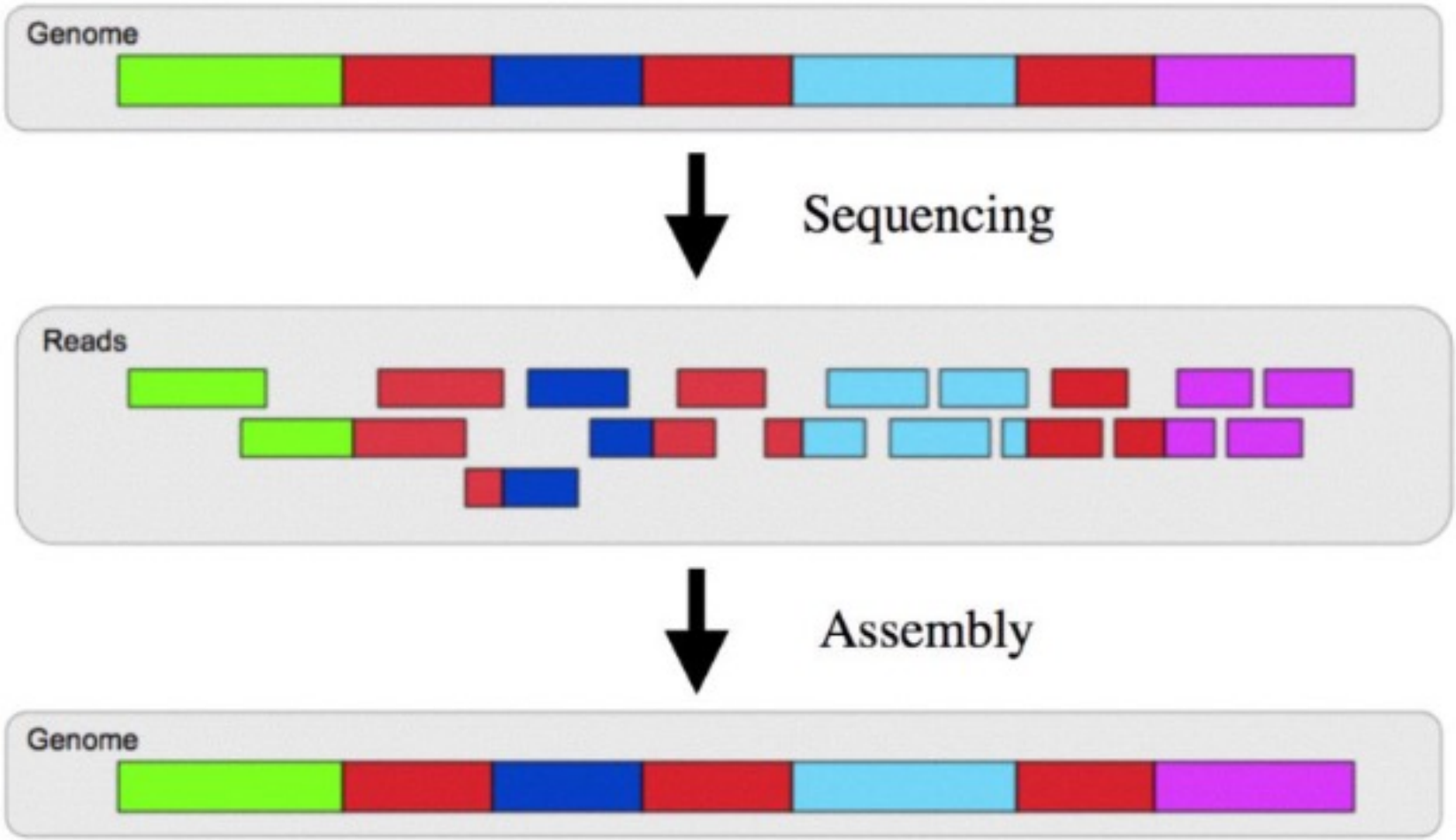
- Here, we use BOWTIE2 to create the index of reference genome.
- Then use it to do the alignment two fastq files with reference genome to create the SAM file.
- Compressed it by SAMTOOLS, create the index which could increase the speed of searching.
- And extract those unmapped reads (only keep those reads not mapped on both paired files).
- Finally convert the BAM file back to two fastq files.

```
-rw-r--r-- 1 xinpeng yinlab         712 Nov 15 19:28 Bowtie2.4401422.err
-rw-r--r-- 1 xinpeng yinlab           0 Nov 15 19:28 Bowtie2.4401422.out
-rw-r--r-- 1 xinpeng yinlab         156 Nov 15 19:28 Bowtie2.4407402.err
-rw-r--r-- 1 xinpeng yinlab           0 Nov 15 19:28 Bowtie2.4407402.out
-rw-r--r-- 1 xinpeng yinlab          91 Nov 15 19:49 Bowtie2.4413614.err
-rw-r--r-- 1 xinpeng yinlab           0 Nov 15 19:33 Bowtie2.4413614.out
-rw-r--r-- 1 xinpeng yinlab 18542547181 Nov 15 19:29 bowtie2_alignments.bam
-rw-r--r-- 1 xinpeng yinlab 90341946317 Nov 15 19:30 bowtie2_alignments.sam
-rw-r--r-- 1 xinpeng yinlab 16001906855 Nov 15 19:31 bowtie2_alignments.sorted.bam
-rw-r--r-- 1 xinpeng yinlab     1227024 Nov 15 19:31 bowtie2_alignments.sorted.bam.bai
-rw-r--r-- 1 xinpeng yinlab         879 Nov 15 19:33 bowtie2.sh
-rw-r--r-- 1 xinpeng yinlab   146782367 Nov 15 19:31 index_prefix.1.bt2
-rw-r--r-- 1 xinpeng yinlab   106750800 Nov 15 19:31 index_prefix.2.bt2
-rw-r--r-- 1 xinpeng yinlab      119672 Nov 15 19:31 index_prefix.3.bt2
-rw-r--r-- 1 xinpeng yinlab   106750795 Nov 15 19:31 index_prefix.4.bt2
-rw-r--r-- 1 xinpeng yinlab   146782367 Nov 15 19:31 index_prefix.rev.1.bt2
-rw-r--r-- 1 xinpeng yinlab   106750800 Nov 15 19:31 index_prefix.rev.2.bt2
-rw-r--r-- 1 xinpeng yinlab 31338813728 Nov 15 19:49 unmapped_1.fastq
-rw-r--r-- 1 xinpeng yinlab 31331098868 Nov 15 19:49 unmapped_2.fastq
-rw-r--r-- 1 xinpeng yinlab 12595279431 Nov 15 19:37 unmapped.bam
[xinpeng@login1.swan new]$ cat Bowtie2.4401422.err

Lmod is automatically replacing "mamba/1.4" with "anaconda/4.12".

116321891 reads; of these:
  116321891 (100.00%) were paired; of these:
    95092684 (81.75%) aligned concordantly 0 times
    8460407 (7.27%) aligned concordantly exactly 1 time
    12768800 (10.98%) aligned concordantly >1 times
    ----
    95092684 pairs aligned concordantly 0 times; of these:
      687366 (0.72%) aligned discordantly 1 time
    ----
    94405318 pairs aligned 0 times concordantly or discordantly; of these:
      188810636 mates make up the pairs; of these:
        183783837 (97.34%) aligned 0 times
        1213045 (0.64%) aligned exactly 1 time
        3813754 (2.02%) aligned >1 times
21.00% overall alignment rate
```

- Finally you will get two fastq files.
- Those are unmapped to the host, i.e., de-contaminated read file.

Part III
Assembly (~ 8hrs)

- We will use cleaned and de-contaminated microbiome reads to re-construct the complete genomes.



- We will try two different programs to make contigs separately (reminder: create two different folders).

- "Megahit": https://github.com/voutcn/megahit
- "MetaSPAdes": https://github.com/ablab/spades

- They are developed by similar algorithms using de bruijn graph.

JOURNAL ARTICLE

MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct *de Bruijn* graph FREE

Dinghua Li, Chi-Man Liu, Ruibang Luo, Kunihiko Sadakane, Tak-Wah Lam ✉ Author Notes

Bioinformatics, Volume 31, Issue 10, May 2015, Pages 1674–1676, https://doi.org/10.1093/bioinformatics/btv033

Published: 20 January 2015 **Article history** ▾

GENOME RESEARCH

CSHL Press | Journal Home | Subscriptions | eTOC Alerts | BioSupplyNet

Genome Res. 2017 May; 27(5): 824–834. PMCID: PMC5411777

doi: 10.1101/gr.213959.116 PMID: 28298430

metaSPAdes: a new versatile metagenomic assembler

Sergey Nurk,[1,4] Dmitry Meleshko,[1,4] Anton Korobeynikov,[1,2] and Pavel A. Pevzner[1,3]

- de bruijn graph: An n-dimensional De Bruijn graph of m symbols is a directed graph representing overlaps between sequences of symbols.

Bridges of Konigsberg problem

AAABBBA

We use k-mer with k=3 as an example

AAABBBA
AAABBBA
AAABBBA
AAABBBA
AAABBBA

AAA is the 3-mer

AAA is the left k-1 mer(2-mer)

AAA is the right k-1 mer(2mer)

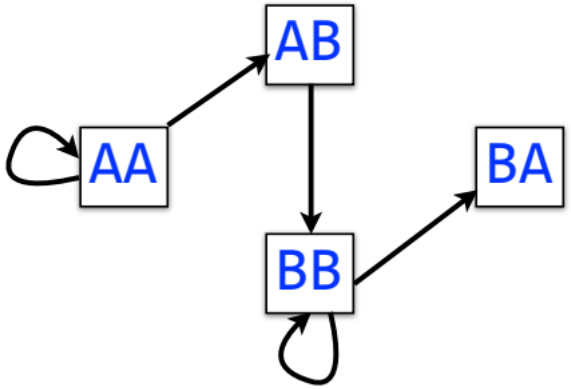
AAA, AAB, ABB, BBB, BBA

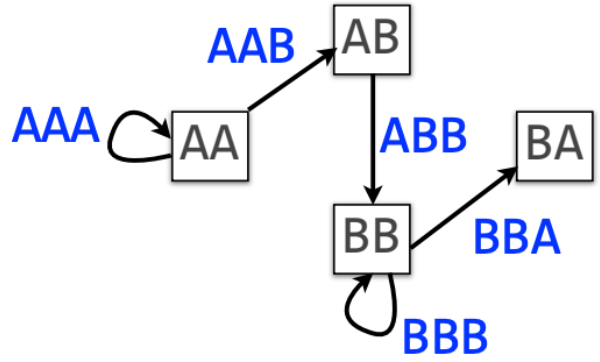
form L/R 2-mers:

AA, AA, AA, AB, AB, BB, BB, BB, BB, BA

L R L R L R L R L R

Each edge in this graph corresponds to a length-3 input string





https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5531759/
https://www.cs.jhu.edu/~langmea/resources/lecture_notes/assembly_dbg.pdf

```
#!/bin/bash
#SBATCH --job-name=megahit
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=32
#SBATCH --time=168:00:00
#SBATCH --mem=250gb
#SBATCH --output=megahit.%J.out
#SBATCH --error=megahit.%J.err
#SBATCH --partition=yinlab,batch,guest


ml megahit/1.2

megahit  -1 your/path/to/your/previous/unmapped_1.fastq -2 your/path/to/your/previous/
unmapped_2.fastq -o megahit_result -t 32
```

```
#!/bin/bash
#SBATCH --job-name=metaspade
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=32
#SBATCH --time=168:00:00
#SBATCH --mem=250gb
#SBATCH --output=metaspade.%J.out
#SBATCH --error=metaspade.%J.err
#SBATCH --partition=yinlab,batch,guest




ml spades/py35/3.13

spades.py --meta -1 unmapped_1.fastq -2 unmapped_2.fastq -o metaspade_result --threads 32
```

- From the MEGAHIT, you will get the final result: final.contigs.fa

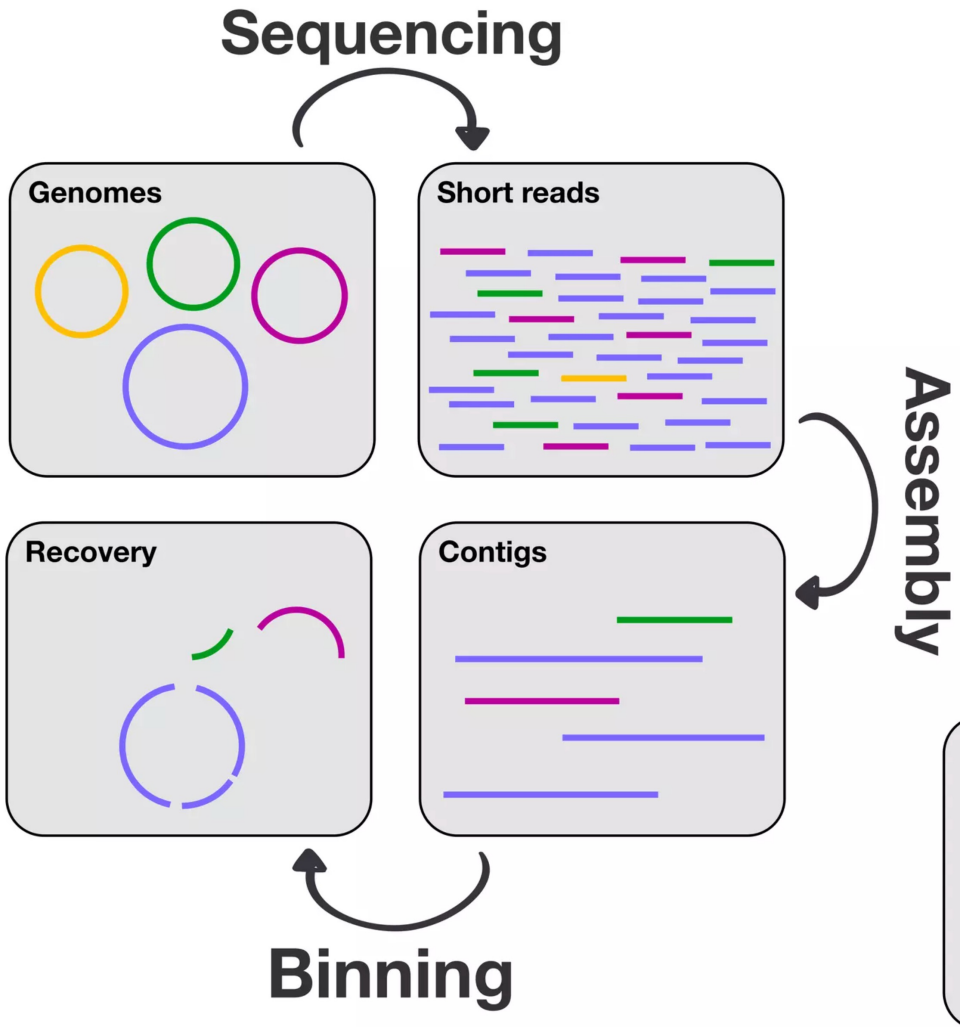
- From MetaSPAde, you will get the final result: contigs.fasta.

- Those are the final results we'll use in the next step.

- In your report, mention the differences between two files, such as the contig count and length (could find in the err/out files).

Part IV
Binning and bin refinement
(~ 2.5hrs)

- binning is the process to group contigs into MAGs.



https://merenlab.org/

- We'll use "MetaWRAP", which contains three different methods.
- https://github.com/bxlab/metaWRAP
- MetaWRAP is a flexible pipeline for genome-resolved metagenomic data analysis. It could finish the metagenomic analysis from start to the end: read quality control, assembly, visualization, taxonomic profiling, extracting draft genomes (binning), and functional annotation(because it includes different tools as packages such as FastQC we used).
- In the future you could use it for your own research, here we'll only use the binning part.

Software | Open access | Published: 15 September 2018

MetaWRAP—a flexible pipeline for genome-resolved metagenomic data analysis

Gherman V. Uritskiy, Jocelyne DiRuggiero & James Taylor

Microbiome **6**, Article number: 158 (2018) | Cite this article

- In the binning part, it includes three different program:

MetaBAT2: https://bitbucket.org/berkeleylab/metabat/src/master/
CONCOCT: https://github.com/BinPro/CONCOCT
MaxBin2: https://github.com/assemblerflow/flowcraft/blob/master/docs/user/components/maxbin2.rst

PeerJ. 2019; 7: e7359. Published online 2019 Jul 26. doi: 10.7717/peerj.7359

PMCID: PMC6662567 | PMID: 31388474

MetaBAT 2: an adaptive binning algorithm for robust and efficient genome reconstruction from metagenome assemblies

Dongwan D. Kang,[1] Feng Li,[2] Edward Kirton,[1] Ashleigh Thomas,[1] Rob Egan,[1] Hong An,[2] and Zhong Wang[1,3,4]

Binning metagenomic contigs by coverage and composition

Johannes Alneberg, Brynjar Smári Bjarnason, Ino de Bruijn, Melanie Schirmer, Joshua Quick, Umer Z Ijaz, Leo Lahti, Nicholas J Loman, Anders F Andersson & Christopher Quince

Nature Methods **11**, 1144–1146 (2014) | Cite this article

JOURNAL ARTICLE

MaxBin 2.0: an automated binning algorithm to recover genomes from multiple metagenomic datasets FREE

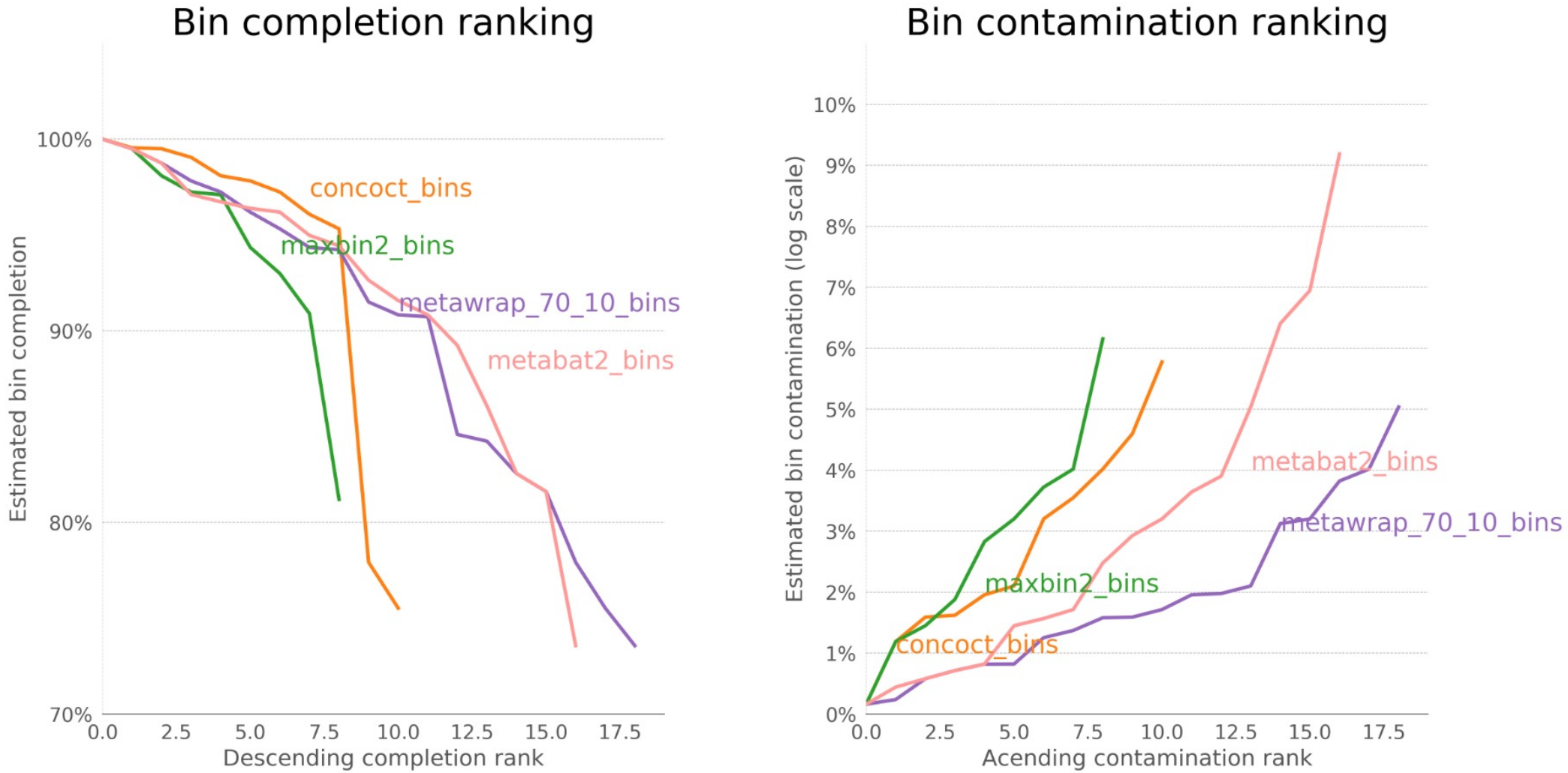
Yu-Wei Wu, Blake A. Simmons, Steven W. Singer Author Notes

Bioinformatics, Volume 32, Issue 4, February 2016, Pages 605–607,
https://doi.org/10.1093/bioinformatics/btv638
Published: 29 October 2015 **Article history**

- In your report, you could discuss the difference among three programs.

- MetaWRAP will combine those three results, give the combinations, for example, A&B, A&C, B&C, A&B&C, and recommend the best result. This step is called "bin refinement".

```
#part4, binning contigs to make bins(MAGs) based on metawrap(~2.5 hrs).

#We'll use metawrap to bin those contigs into bins which is the metagenome assembled
genomes (MAGs).
#Remember you need to do both metaSPAde result and Megahit result.

#!/bin/bash
#SBATCH --job-name=metawrap
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=16
#SBATCH --time=168:00:00
#SBATCH --mem=250gb
#SBATCH --output=metawrap.%J.out
#SBATCH --error=metawrap.%J.err
#SBATCH --partition=yinlab,batch,guest

ml metawrap/1.3

metawrap binning -o BINNING_megahit -t 16 -a your_assembly_contig_fasta_file --metabat2 --
maxbin2 --concoct your_fastq1file_removed_algae_genes your_fastq2file_removed_algae_genes
-t 16

metawrap bin_refinement -o BIN_REFINEMENT_megahit -t 32 -A BINNING_megahit/metabat2_bins/
-B BINNING_megahit/maxbin2_bins/ -C BINNING_megahit/concoct_bins/

#In the metawrap, it combines three different method to do the bins, and bin_refinement is
to combine those results together and summarize to give the final result.

#Compare those results and report.
#the final results are in BIN_REFINEMENT_megahit/metawrap_70_10_bins/*.fa, each of those
means a MAG.
```



- You will find this figure in theresult folder.
- Results are in BIN_REFINEMENT_megahit/metawrap_70_10_bins/*.fa, each is a MAG.

Part V
Bin quality check with CheckM2
(~ 30mins)

- The output of bin_refinement are: BIN_REFINEMENT_megahit/metawrap_70_10_bins/*.fa.

- Those fa files are the MAGs we need. Next we‘ll check the quality and contamination(for example, if it contains wrong reads from other bacteria) of those MAGs.

- MetaWRAP includes the program CheckM (default is completeness >70.0%, contamination <10.0%, that‘s the output folder name, metawrap_70_10_bins).

- CheckM was updated to CheckM2

GENOME RESEARCH
CSHL Press | Journal Home | Subscriptions | eTOC Alerts | BioSupplyNet

Genome Res. 2015 Jul; 25(7): 1043–1055. doi: 10.1101/gr.186072.114

PMCID: PMC4484387 | PMID: 25977477

CheckM: assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes

Donovan H. Parks,[1] Michael Imelfort,[1] Connor T. Skennerton,[1] Philip Hugenholtz,[1,2] and Gene W. Tyson[1,3]

Article | Published: 27 July 2023

CheckM2: a rapid, scalable and accurate tool for assessing microbial genome quality using machine learning

Alex Chklovski, Donovan H. Parks, Ben J. Woodcroft & Gene W. Tyson

Nature Methods **20**, 1203–1212 (2023) | Cite this article

- So we'll run the new version CheckM2 to check the quality again.
- HCC has not installed it because it's really new (that's the reason why we will use conda).

```
#part5, check the quality of bins by CheckM2.
#checkM is a famous program to check the quality of assembled genomes, and they update it as CheckM2 this
year.
#on hcc they have not installed it, so we'll use conda to install it.
#don't forget to do both metaSPAde and Megahit

mkdir part5_checkm2
cd part5_checkm2
git clone --recursive https://github.com/chklovski/checkm2.git && cd checkm2
conda env create -n checkm2 -f checkm2.yml
conda activate checkm2
bin/checkm2 -h #to check if it works
checkm2 database --download --path /custom/path/
export CHECKM2DB="path/to/database"
```

```
#!/bin/bash
#SBATCH --job-name=checkm2
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=16
#SBATCH --time=168:00:00
#SBATCH --mem=250gb
#SBATCH --output=checkm2.%J.out
#SBATCH --error=checkm2.%J.err
#SBATCH --partition=vinlab,batch,guest

ml anaconda

conda activate checkm2


your/onw/path/bin/checkm2  predict --threads 16 --input ../bining/BIN_REFINEMENT_megahit/
metawrap_70_10_bins/*.fa  --output-directory checkm2_result
```

https://github.com/chklovski/CheckM2

```
[xinpeng@login1.swan checkm2_result]$ cat quality_report.tsv
Name    Completeness    Contamination   Completeness_Model_Used Translation_Table_Used  Coding_Density  Contig_N50      Average_Gene
_Length Genome_Size     GC_Content      Total_Coding_Sequences  Total_Contigs   Max_Contig_Length       Additional_Notes
bin.1   94.12   1.26    Neural Network (Specific Model) 11      0.894   208719  316.9761048847536       5022228 0.67    4729    37 5
30019   None
bin.10  83.7    3.92    Neural Network (Specific Model) 11      0.882   141213  308.18683239463974      5391826 0.66    5149    71 6
21662   None
```

- In your report, you should report the whole table of this result.

- In the MIMAG paper (see links below), they recommend "more than 90% complete and have less than 5% contamination" as the high-quality MAGs. Report which are the high-quality MAGs.

https://www.nature.com/articles/nbt.3893

Perspective | Open access | Published: 01 August 2017

Minimum information about a single amplified genome (MISAG) and a metagenome-assembled genome (MIMAG) of bacteria and archaea

Robert M Bowers, Nikos C Kyrpides, Ramunas Stepanauskas, Miranda Harmon-Smith, Devin Doud, T B K Reddy, Frederik Schulz, Jessica Jarett, Adam R Rivers, Emiley A Eloe-Fadrosh, Susannah G Tringe, Natalia N Ivanova, Alex Copeland, Alicia Clum, Eric D Becraft, Rex R Malmstrom, Bruce Birren, Mircea Podar, Peer Bork, George M Weinstock, George M Garrity, Jeremy A Dodsworth, Shibu Yooseph, Granger Sutton, The Genome Standards Consortium, ... Tanja Woyke + Show authors

Nature Biotechnology **35**, 725–731 (2017) | Cite this article

Part VI
Taxonomy annotation
(~ 1-2hrs)

- Answer the question: what taxonomic groups do my MAGs belong to?
- We'll use the bins(MAGs) to do the taxonomy annotation by "GTDBtk".
- GTDB(Genome Taxonomy Database) is a very famous database for taxonomy, and the tool GTDBtk is to do the tax annotation based on that.

JOURNAL ARTICLE

GTDB-Tk v2: memory friendly classification with the genome taxonomy database

Pierre-Alain Chaumeil, Aaron J Mussig, Philip Hugenholtz, Donovan H Parks

Bioinformatics, Volume 38, Issue 23, 1 December 2022, Pages 5315–5316,
https://doi.org/10.1093/bioinformatics/btac672
Published: 11 October 2022 **Article history**

JOURNAL ARTICLE

GTDB: an ongoing census of bacterial and archaeal diversity through a phylogenetically consistent, rank normalized and complete genome-based taxonomy

Donovan H Parks, Maria Chuvochina, Christian Rinke, Aaron J Mussig, Pierre-Alain Chaumeil, Philip Hugenholtz

Nucleic Acids Research, Volume 50, Issue D1, 7 January 2022, Pages D785–D794,
https://doi.org/10.1093/nar/gkab776
Published: 14 September 2021 **Article history**

- Here we use all the MAGs (70% comp, 10 cont), but in real project, people often only do it for high quality MAGs.

```
#!/bin/bash
#SBATCH --job-name=gtdb
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=16
#SBATCH --time=168:00:00
#SBATCH --mem=250gb
#SBATCH --output=gtdb.%J.out
#SBATCH --error=gtdb.%J.err
#SBATCH --partition=vinlab,batch,guest

ml gtdbtk/1.5

gtdbtk classify_wf --genome_dir ../bining/BIN_REFINEMENT_megahit/metawrap_70_10_bins/ --out_dir GTDB/ --cpus
16 --extension .fa
```

```
[xinpeng@login1.swan GTDB]$ less -S gtdbtk.bac120.summary.tsv
```

```
user_genome     classification  fastani_reference       fastani_reference_radius        fastani_taxonomy        fastani_ani     fas>
bin.1   d__Bacteria;p__Proteobacteria;c__Alphaproteobacteria;o__Rhizobiales;f__Beijerinckiaceae;g__Bosea;s__Bosea vestrisii     GCF>
bin.10  d__Bacteria;p__Proteobacteria;c__Alphaproteobacteria;o__Rhizobiales;f__Xanthobacteraceae;g__Bradyrhizobium;s__  N/A     N/A>
bin.11  d__Bacteria;p__Proteobacteria;c__Alphaproteobacteria;o__Rhizobiales;f__Xanthobacteraceae;g__Afipia;s__  N/A     N/A     N/A>
bin.12  d__Bacteria;p__Proteobacteria;c__Alphaproteobacteria;o__Rhizobiales;f__Rhizobiaceae;g__Mesorhizobium;s__Mesorhizobium terra>
bin.13  d__Bacteria;p__Proteobacteria;c__Alphaproteobacteria;o__Rhizobiales;f__Xanthobacteraceae;g__Afipia;s__Afipia birgiae    GCF>
bin.14  d__Bacteria;p__Proteobacteria;c__Gammaproteobacteria;o__Nevskiales;f__Nevskiaceae;g__Nevskia;s__        N/A     N/A     N/A>
bin.15  d__Bacteria;p__Proteobacteria;c__Alphaproteobacteria;o__Rhizobiales;f__Rhizobiaceae;g__Rhizobium;s__    N/A     N/A     N/A>
bin.16  d__Bacteria;p__Actinobacteriota;c__Actinomycetia;o__Mycobacteriales;f__Mycobacteriaceae;g__Mycobacterium;s__    N/A     N/A>
bin.17  d__Bacteria;p__Bacteroidota;c__Bacteroidia;o__Chitinophagales;f__Chitinophagaceae;g__Puia;s__Puia sp001898505   GCA_0018985>
bin.18  d__Bacteria;p__Proteobacteria;c__Alphaproteobacteria;o__Sphingomonadales;f__Sphingomonadaceae;g__Sphingomonas;s__Sphingomon>
bin.19  d__Bacteria;p__Acidobacteriota;c__Acidobacteriae;o__Bryobacterales;f__Bryobacteraceae;g__;s__   N/A     N/A     N/A     N/A>
bin.2   d__Bacteria;p__Proteobacteria;c__Alphaproteobacteria;o__Rhizobiales;f__Rhizobiaceae;g__Mesorhizobium;s__Mesorhizobium sp005>
bin.3   d__Bacteria;p__Proteobacteria;c__Gammaproteobacteria;o__Burkholderiales;f__Burkholderiaceae;g__Pelomonas;s__    N/A     N/A>
bin.4   d__Bacteria;p__Proteobacteria;c__Gammaproteobacteria;o__Burkholderiales;f__Burkholderiaceae;g__Cupriavidus;s__Cupriavidus b>
bin.5   d__Bacteria;p__Bacteroidota;c__Bacteroidia;o__Sphingobacteriales;f__Sphingobacteriaceae;g__Mucilaginibacter;s__Mucilaginiba>
bin.6   d__Bacteria;p__Proteobacteria;c__Alphaproteobacteria;o__Rhizobiales;f__Xanthobacteraceae;g__Z2-YC6860;s__       N/A     N/A>
bin.7   d__Bacteria;p__Proteobacteria;c__Alphaproteobacteria;o__Reyranellales;f__Reyranellaceae;g__Reyranella;s__       N/A     N/A>
bin.8   d__Bacteria;p__Proteobacteria;c__Alphaproteobacteria;o__Sphingomonadales;f__Sphingomonadaceae;g__Sphingomonas;s__Sphingomon>
bin.9   d__Bacteria;p__Proteobacteria;c__Gammaproteobacteria;o__Burkholderiales;f__Burkholderiaceae;g__Pandoraea;s__Pandoraea norim>
gtdbtk.bac120.summary.tsv (END)
```

- Create a table to report this result.
- For each bin, check the details of taxonomy and report.

Part VII
Gene prediction and functional annotation
(~ 8hrs)

- Answer the question: what functional genes do my MAGs encode?

- First we need predict proteins by genes from MAGs.

- Called gene prediction.

- Then annotate the function of each protein.
- Called gene annotation.

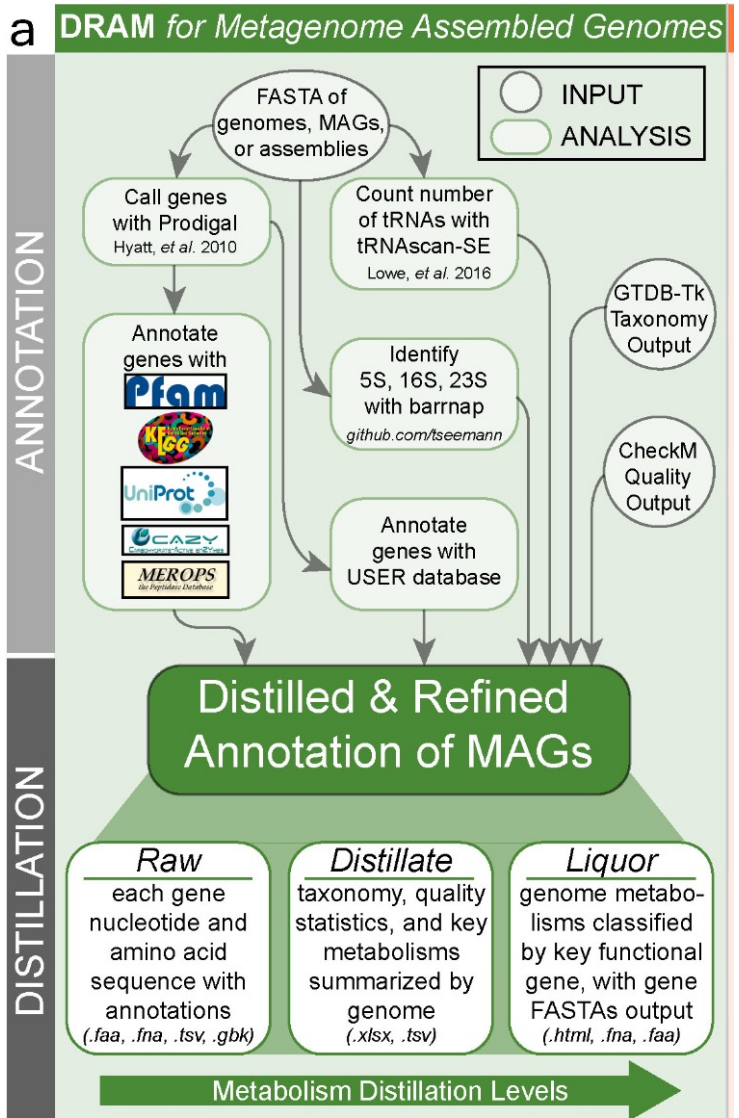
- We'll use "DRAM", which does both.
- DRAM integrates different tools for both steps including 7 different databases (including dbCAN) for functional annotation (focus on metabolic enzymes).

Nucleic Acids Research

Nucleic Acids Res. 2020 Sep 18; 48(16): 8883–8900. PMCID: PMC7498326
Published online 2020 Aug 7. doi: 10.1093/nar/gkaa621 PMID: 32766782

DRAM for distilling microbial metabolism to automate the curation of microbiome function

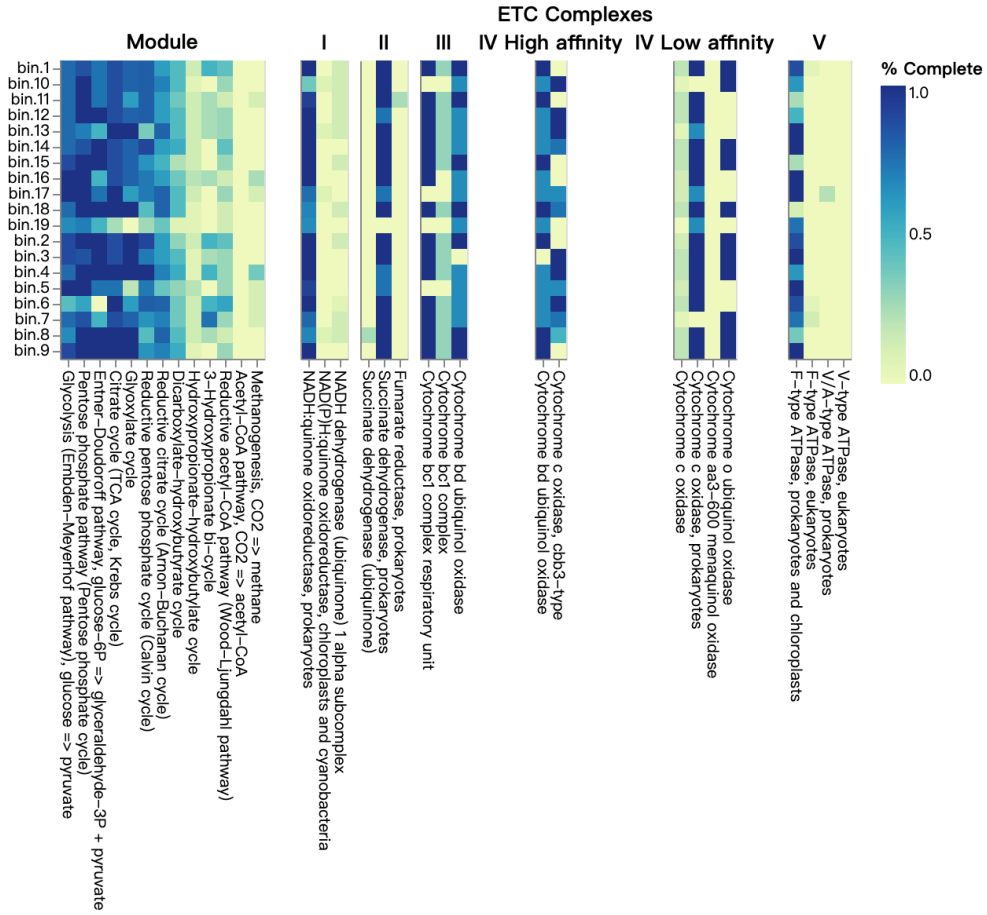
Michael Shaffer, Mikayla A Borton, Bridget B McGivern, Ahmed A Zayed, Sabina Leanti La Rosa, Lindsey M Solden, Pengfei Liu, Adrienne B Narrowe, Josué Rodríguez-Ramos, Benjamin Bolduc, M Consuelo Gazitúa, Rebecca A Daly, Garrett J Smith, Dean R Vik, Phil B Pope, Matthew B Sullivan, Simon Roux, and Kelly C Wrighton

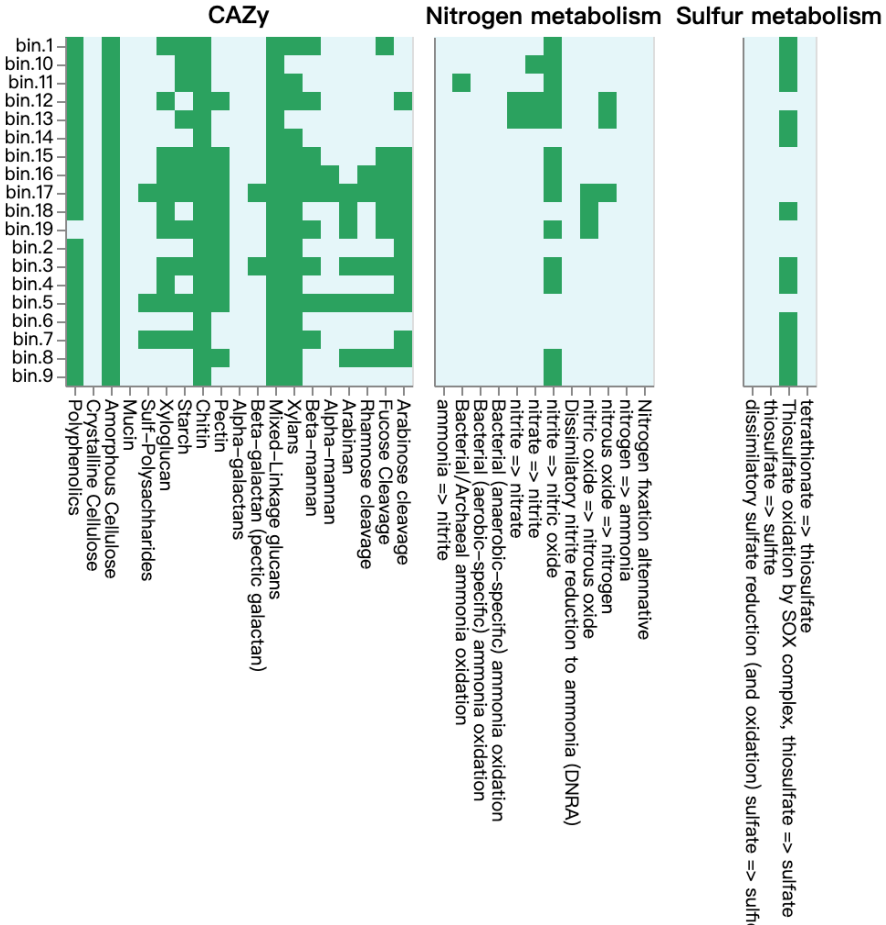


https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7498326/
https://github.com/WrightonLabCSU/DRAM/wiki

```
#!/bin/bash
#SBATCH --job-name=dram
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=16
#SBATCH --time=168:00:00
#SBATCH --mem=250gb
#SBATCH --output=dram.%J.out
#SBATCH --error=dram.%J.err
#SBATCH --partition=yinlab,batch,guest

ml dram/1.2

DRAM.py annotate -i '../bining/BIN_REFINEMENT_megahit/metawrap_70_10_bins/*.fa' (do not delete the quotation
marks)  -o annotation
DRAM.py distill -i annotation/annotations.tsv -o distill --trna_path annotation/trnas.tsv --rrna_path
annotation/rrnas.tsv
```

- You would get two heatmaps for metabolism.
- Report and explain the results(html file) combined with your taxonomy annotation for each MAG.





Summary

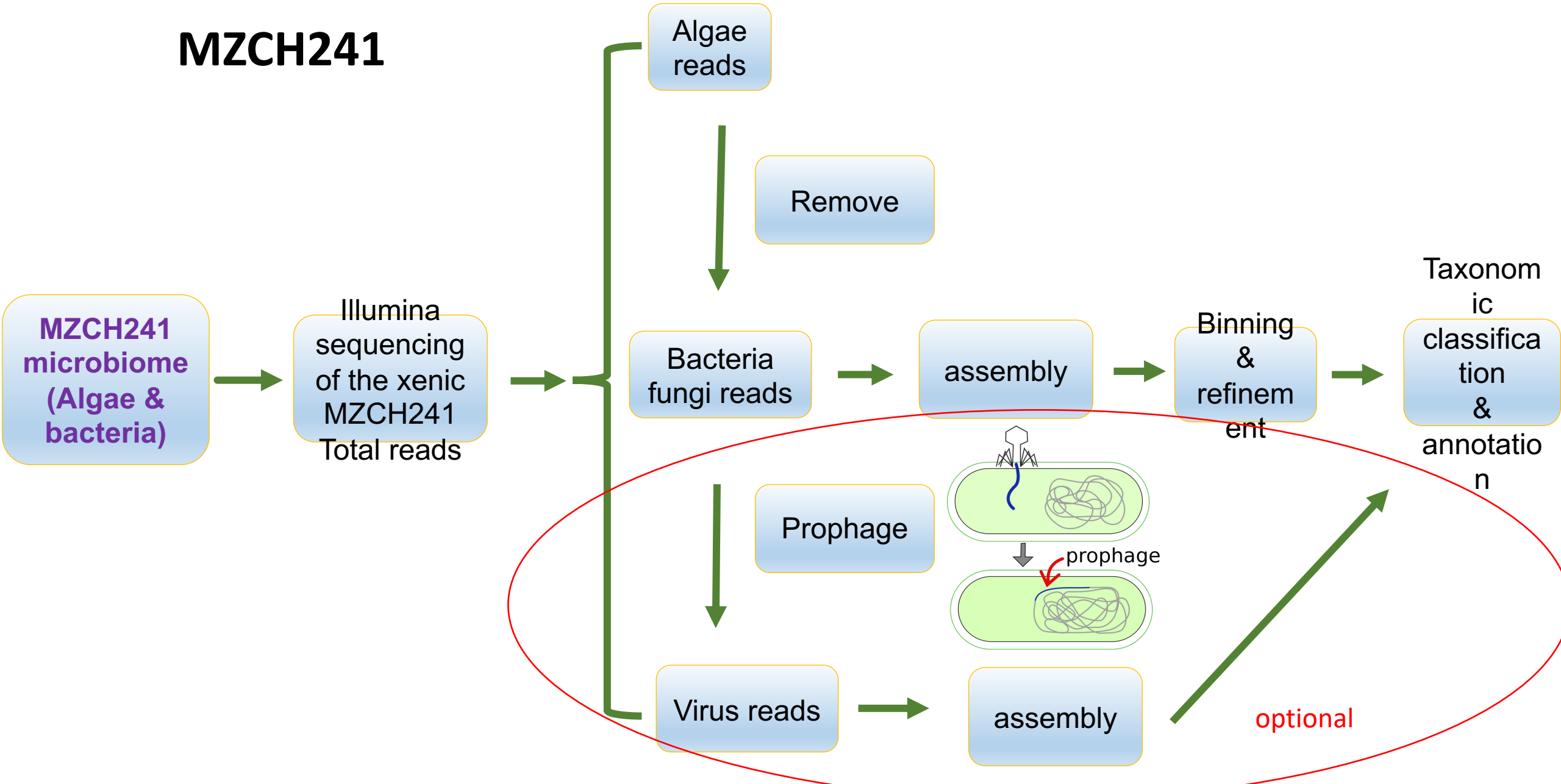
- We reconstructed MAGs based on raw reads.
- For each part you need a result figure and table (if has).
- Report the questions and discussion we mentioned in the slides.

Sub-project II. Virome analysis in algal microbiome data (optional)

1. Find virome contigs
2. Taxonomy classification
3. Host prediction
4. Function and AMG (Auxiliary metabolic gene) annotation.

Virome workflow of *Zygnema circumcarinatum*

MZCH241

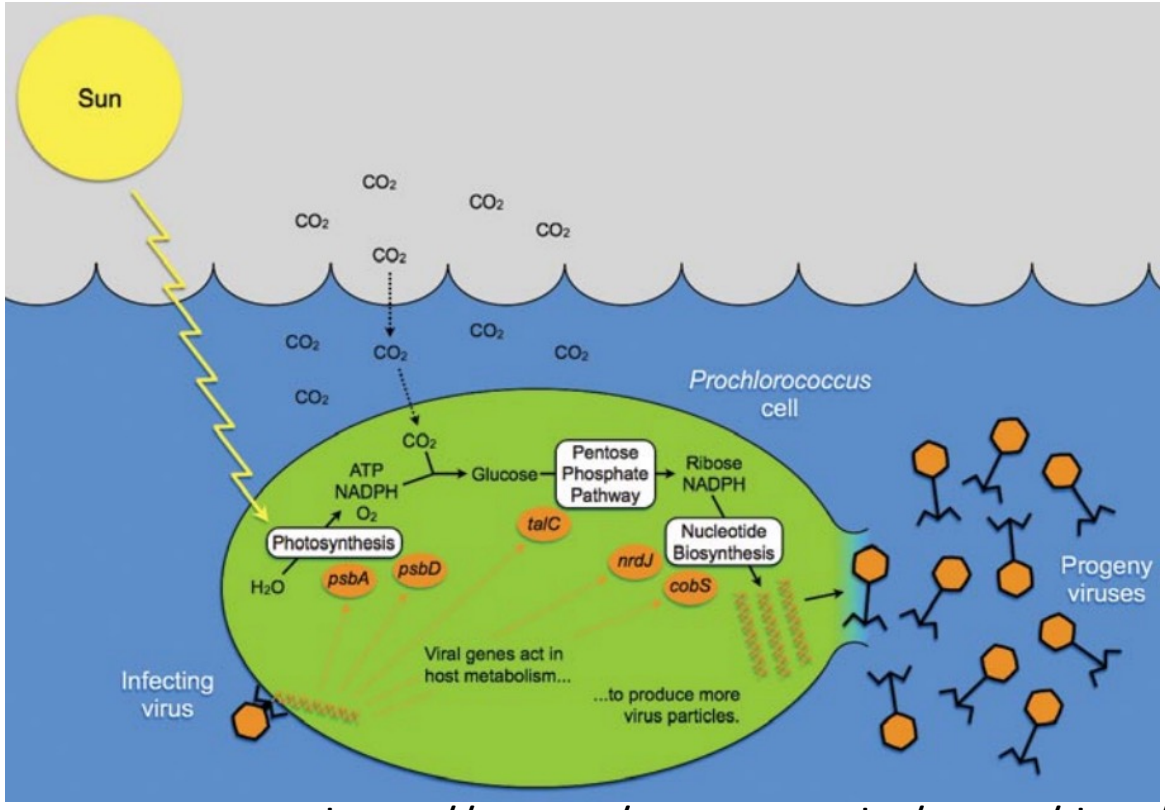


•Viruses (virome) contribute part of the metagenome reads (5.8% of total microbial DNAs in human gut are from phages).

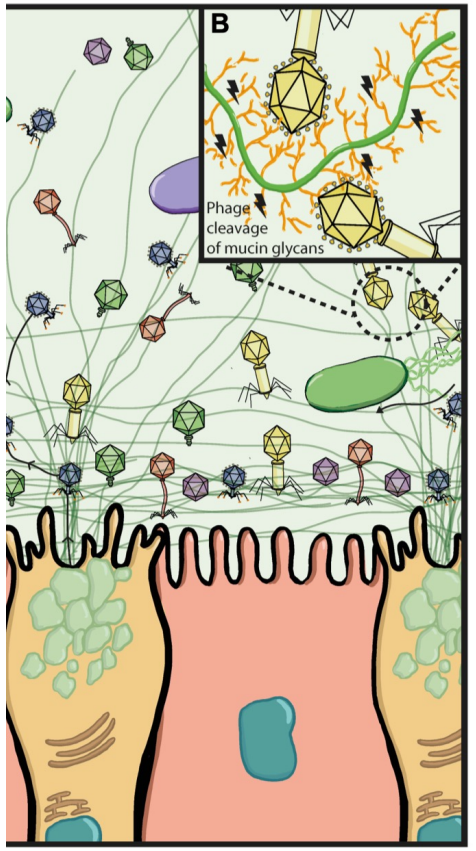
•About 25% of all phages on Earth exist as prophages (Roshan et al., 2023) providing around 20% of genomes in bacteria.

•Some phages carry host-derived enzymes (AMGs), which can help the host with metabolic activities

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5380687/



https://tos.org/oceanography/assets/docs/20-2_breitbart.pdf



Part 0
Program preparation
(~5 mins)

- We'll use "VirSorter2" to find the viral contigs from the total contigs assembled from microbiome reads.
- Because the version of hcc is old, we'll use conda/mamba to install the new version.

```
conda create -n viral-id-sop virsorter=2
virsorter setup -d db-vs2 -j 4
```

Software article | Open access | Published: 01 February 2021

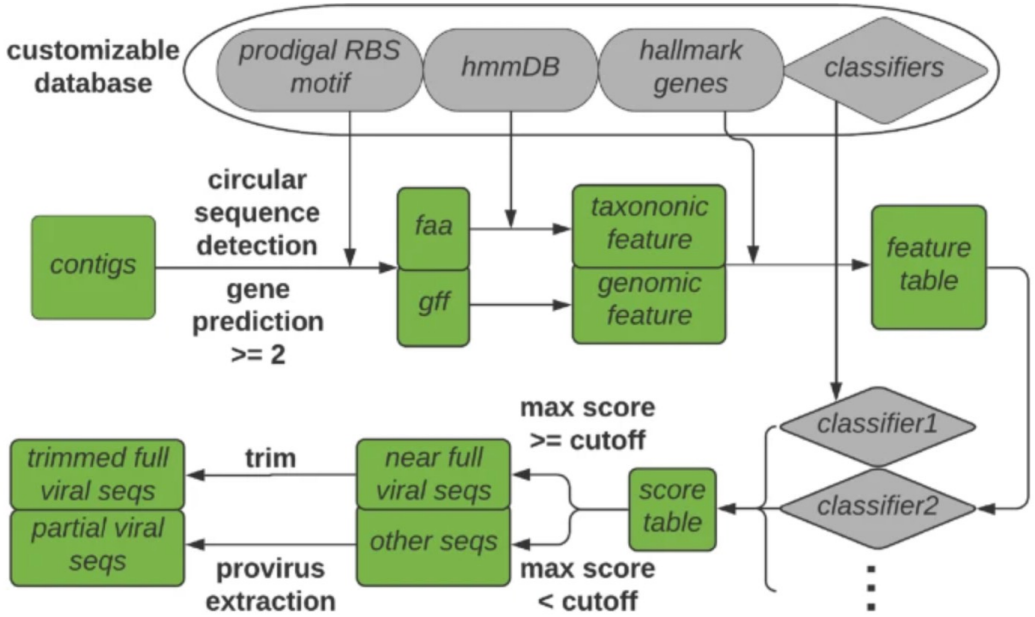
VirSorter2: a multi-classifier, expert-guided approach to detect diverse DNA and RNA viruses

Jiarong Guo, Ben Bolduc, Ahmed A. Zayed, Arvind Varsani, Guillermo Dominguez-Huerta, Tom O. Delmont, Akbar Adjie Pratama, M. Consuelo Gazitúa, Dean Vik, Matthew B. Sullivan & Simon Roux

Microbiome **9**, Article number: 37 (2021) | Cite this article

25k Accesses | **265** Citations | **79** Altmetric | Metrics

https://microbiomejournal.biomedcentral.com/articles/10.1186/s40168-020-00990-y



Part I
Viral contig search and function annotation
(~ 30mins – 1hrs)

- We will combine multiple commands in one slurm script (explained in next slide).

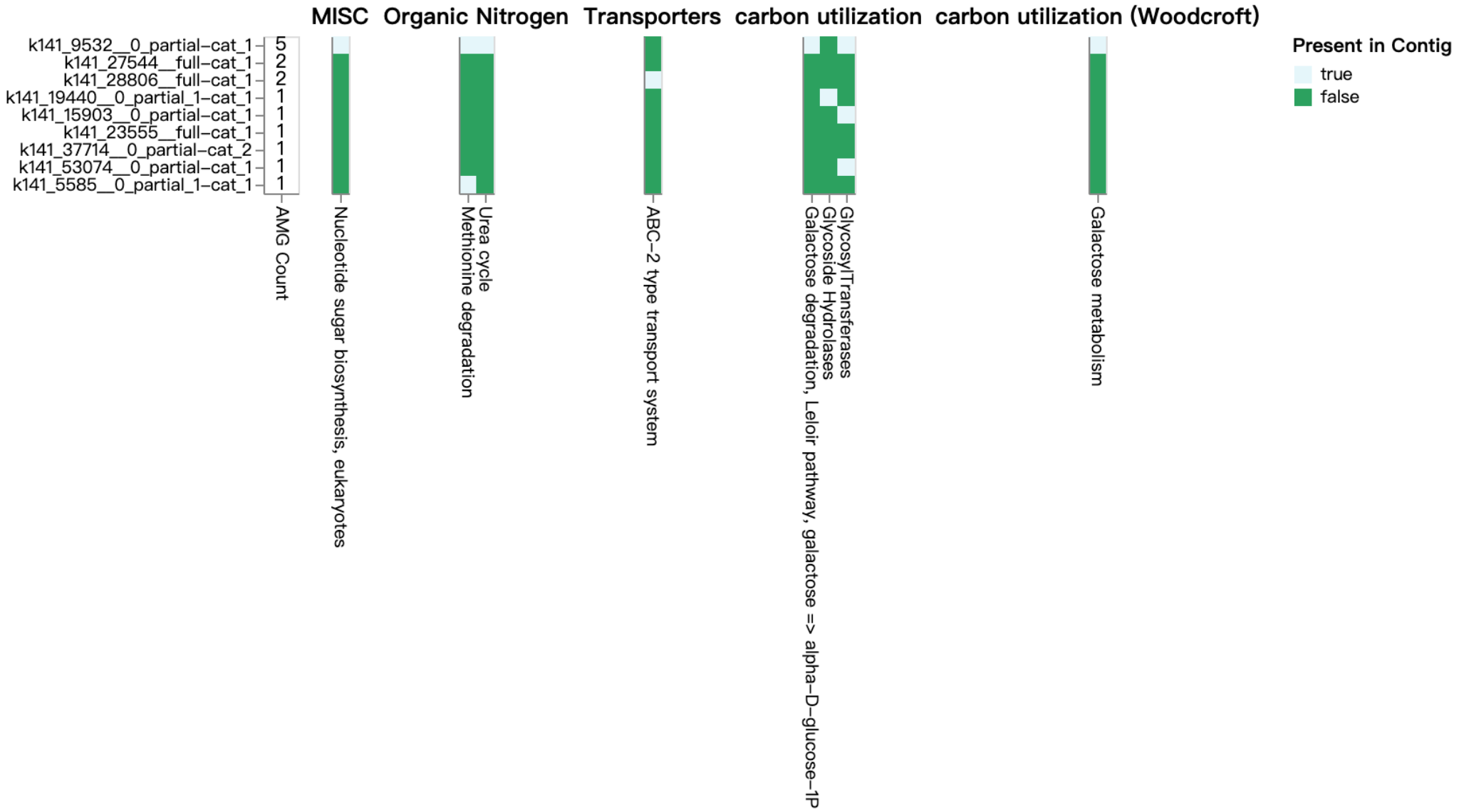
```
#!/bin/bash
#SBATCH --job-name=virsorter_1
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=32
#SBATCH --time=168:00:00
#SBATCH --mem=250gb
#SBATCH --output=virsorter2.%J.out
#SBATCH --error=virsorter2.%J.err
#SBATCH --partition=yinlab,batch,guest

ml anaconda
conda activate viral-id-sop


virsorter run --keep-original-seq -i your_contig_fa_from_megahit_or_metaSPAde -w vs2-pass1 --include-groups
dsDNAphage,ssDNA --min-length 5000 --min-score 0.5 -j 32 all

ml checkv

checkv end_to_end vs2-pass1/final-viral-combined.fa checkv -t 28 -d /fs/project/PAS1117/jiarong/db/checkv-db-v1.0


virsorter run --seqname-suffix-off --viral-gene-enrich-off --provirus-off --prep-for-dramv -i checkv/combined.fna -w
vs2-pass2 --include-groups dsDNAphage,ssDNA --min-length 5000 --min-score 0.5 -j 32 all

ml dram/1.2

DRAM-v.py annotate -i vs2-pass2/for-dramv/final-viral-combined-for-dramv.fa -v vs2-pass2/for-dramv/viral-affi-
contigs-for-dramv.tab -o dramv-annotate --skip_trnascan --threads 32 --min_contig_size 1000

DRAM-v.py distill -i dramv-annotate/annotations.tsv -o dramv-distill
```

- After virsorter2, we'll run a program called "CheckV", which is the virus version of CheckM.

- Using CheckV we could filter the results of VirSorter2 and give us more confident viral contigs.

- And then use the result of CheckV to run VirSorter2 again to prepare the input file of DRAM-v, which is the viral version of DRAM.

- Finally, we'll use DRAM-v for the function annotation and find AMGs.
  In dramv-distill folder, you could find a visualization of AMGs.



- Report this result and explain it.

Part II
Further filtering of VirSorter contigs

- It's hard to distinguish viral contigs and cellular contigs

https://www.protocols.io/view/viral-sequence-identification-sop-with-virsorter2-5qpvoyqebg4o/v3?step=3

- Based on our benchmark with a soil bulk metagenome, those with no viral and no host gene called are viral; those with no viral gene and 2 or more host genes are mostly non-viral; those with no viral gene and 1 host gene are hard to tell viral from non-viral, and should be discarded unless manually checked.

- Here we only select those >10kb for manual curation since shorter ones are too short to tell.

- Also those with VirSorter2 score ≥0.95 or hallmark gene count >2 are mostly viral. These empirical screening criteria are summarized below:

•**Keep1:** viral_gene >0
•**Keep2**: viral_gene =0 AND (host_gene =0 OR score >=0.95 OR hallmark >2)

```
#we only keep those two situation, keep 1 could be gotten from checkv/contamination.tsv

cat checkv/contamination.tsv | awk '{if($4>0) print $1}' > keep1
```

```
#keep 2 could extract viral_genes=0 and then keep this list to map in the vs2-pass1/final-viral-score.tsv.

cat checkv/contamination.tsv | awk '{if($4==0) print $0}' | awk '{if($5 ==0) print $1}' > keep2_1
cat checkv/contamination.tsv | awk '{if($4==0) print $1}' > keep2_potential
grep -wFf keep2_potential vs2-pass1/final-viral-score.tsv | awk '{if ($4 >= 0.95) print$1}' > keep2_2
grep -wFf keep2_potential vs2-pass1/final-viral-score.tsv | awk '{if ($7 > 2) print$1}' > keep2_3
```

- Combine the keep1 and keep2 into the final result by "cat".
- Use "seqkit" to extract the high-confident viral contigs.
- Report each amount of keep, and the total situation.

```
ml seqkit
cat keep1 keep2_1 keep2_2 keep2_3 > total_selected
seqkit grep -n -f total_selected vs2-pass1/final-viral-combined.fa > potentail_virus.fa
```

Part II
Phage gene prediction and annotation (~30 mins)

- In DRAM-v, it uses prodigal to do the phage gene prediction and annotation. However, prodigal is designed for bacteria genes, although it's also used in phages.
- We'll use another program called Pharokka which is specifically designed for phages.
- In your report, you could compare the result of Pharokka and DRAM-v.

JOURNAL ARTICLE

Pharokka: a fast scalable bacteriophage annotation tool

George Bouras ✉, Roshan Nepal, Ghais Houtak, Alkis James Psaltis, Peter-John Wormald, Sarah Vreugde

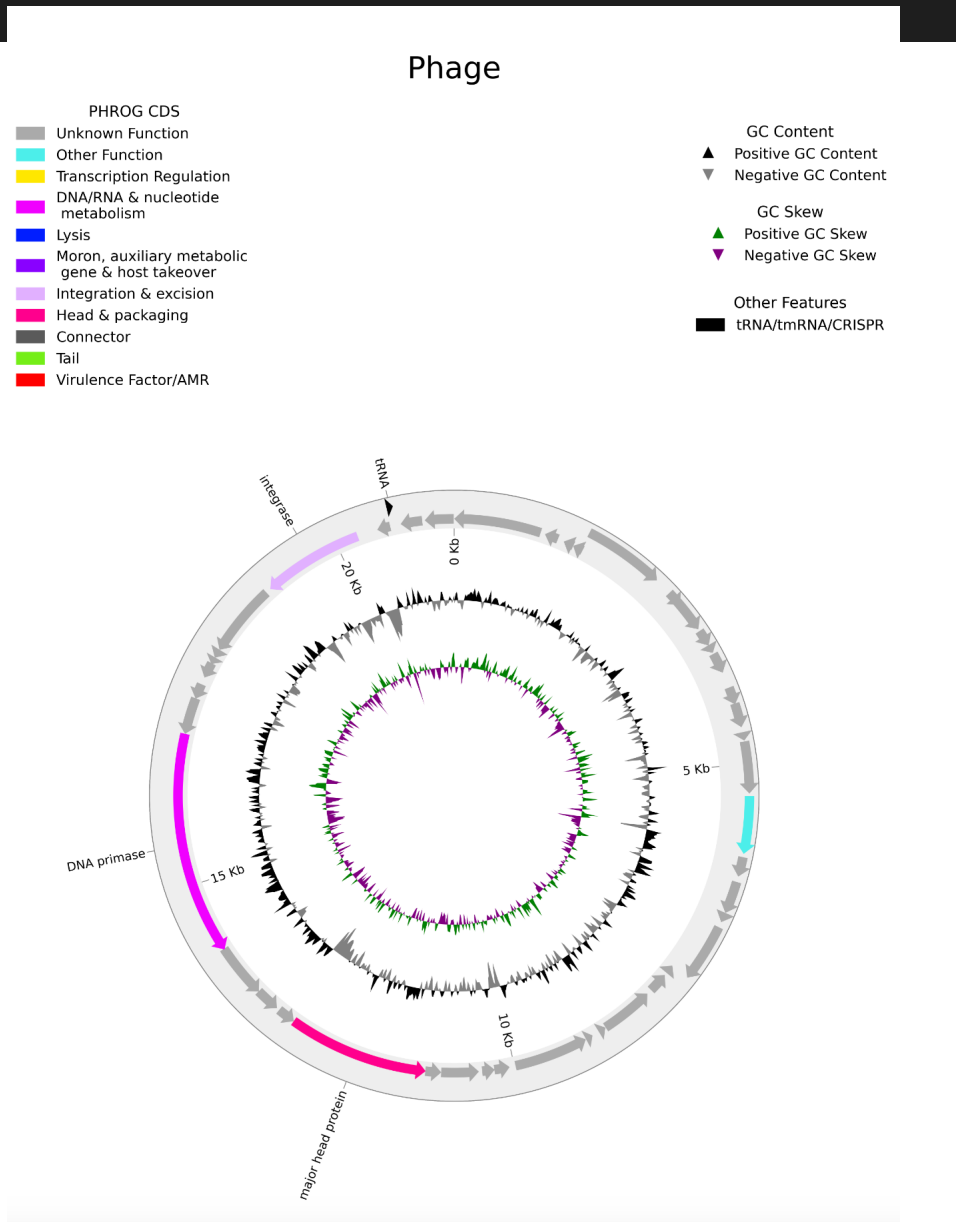
Bioinformatics, Volume 39, Issue 1, January 2023, btac776,
https://doi.org/10.1093/bioinformatics/btac776
Published: 01 December 2022 **Article history** ▾

```
git clone https://github.com/gbouras13/pharokka.git
cd pharokka
mamba env create -f environment.yml -p $COMMON/conda_env/pharokka
conda activate /common/vinlab/xinpeng/conda_env/pharokka
pharokka/bin/install_databases.py

#it will tell you the path of db
```

```
#!/bin/bash
#SBATCH --job-name=pharokka
#SBATCH --ntasks-per-node=32
#SBATCH --time=168:00:00
#SBATCH --mem=100gb
#SBATCH --output=pharokka.%J.out
#SBATCH --error=pharokka.%J.err
#SBATCH --partition=vinlab,batch,guest
#SBATCH --license=common


ml anaconda
conda activate /common/vinlab/xinpeng/conda_env/pharokka

pharokka/bin/pharokka.py -i ../virsorter2/potentail_virus.fa -o annotation -d /work/vinlab/xinpeng/
final_course_project/new/preprocessing/tax_virus/pharokka/databases -t 32
```

```
ml seqkit
seqkit split -i -f -O split_contig_for_plot --by-id ../virsorter2/potentail_virus.fa
for i in `ls split_contig_for_plot`; do pharokka/bin/pharokka_plotter.py -i $1 -n $i -o pharokka_output_directory;
done
```

- For each viral contig, we could extract from the total viral contig, and plot those contigs with gene annotation.
- You may report this result in your report.



Part III
Phage taxonomy classification and host prediction
(~ 1hr)

- Viral contigs are difficult to taxonomically classify because:

1.There are no universally accepted marker genes for virus taxonomy.

2. Short genomes and rapid evolution.

Need to use a different approach for taxonomy classification; however, most phages are still taxonomically unassigned.

We'll use vCONTACT2, a clustering program for viral taxonomy using shared gene content.

```
#After we get the protein sequence file, we'll use vCONTACT2 to do the tax prediction.
#for vCONTACT2, it needs two input file:
sed '/#/d' annotation/pharokka.gff | grep "ID=" | cut -f 9 | cut -d';' -f1 > proteinid

sed '/#/d' annotation/pharokka.gff | grep "ID=" | cut -f 1 > contigid

paste -d',' proteinid contigid  > gene2genome.csv
sed -i 's/$/,None_provided/' gene2genome.csv
sed -i 's/ID=//' gene2genome.csv
sed -i '1s/^/protein_id,contig_id,keywords\n/' gene2genome.csv
```

```
#!/bin/bash
#SBATCH --job-name=vcontact2
#SBATCH --ntasks-per-node=32
#SBATCH --time=168:00:00
#SBATCH --mem=100gb
#SBATCH --output=vcontact2.%J.out
#SBATCH --error=vcontact2.%J.err
#SBATCH --partition=yinlab,batch,guest
#SBATCH --license=common


ml vcontact2
vcontact2 --raw-proteins you_protein_file --proteins-fp proteins --db 'ProkaryoticViralRefSeq85-Merged' --output-dir
vcontact2_output --threads 32
```

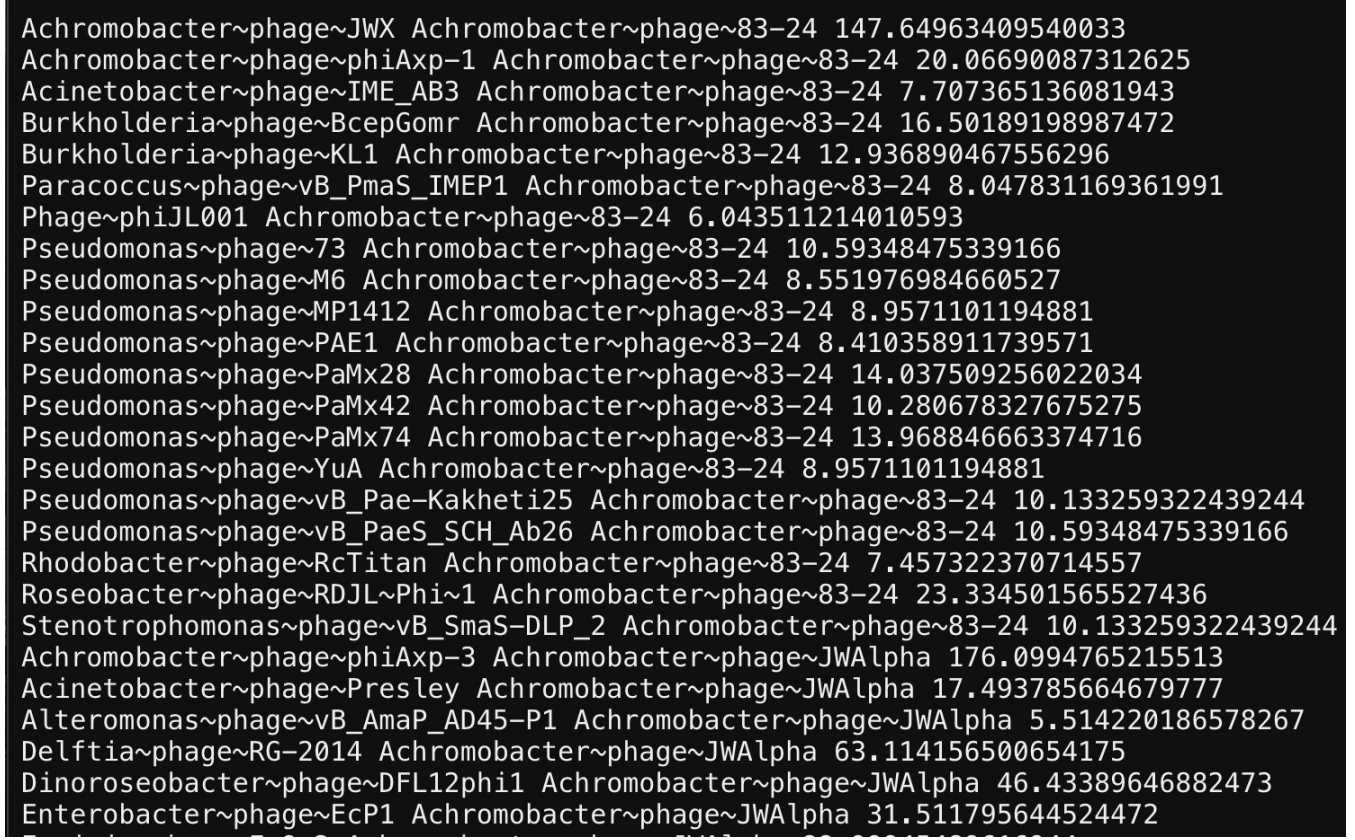
vCONTACT2 achieves TAXONOMY CLASSIFICATION of viral contigs by clustering them with refseq viral genomes (with known VIRAL taxonomy); the resultant files can be imported into Cytoscape for plotting.

```
[xinpeng@login1.swan vcontact2_output]$ head genome_by_genome_overview.csv
,Genome,Order,Family,Genus,VC,VC Status,Size,VC Subcluster,VC Subcluster Size,Quality,Adj P-value,Topology Confidence Score,Genera i
n VC,Families in VC,Orders in VC,Genus Confidence Score
0,Achromobacter~phage~83-24,Caudovirales,Siphoviridae,Jwxvirus,0_0,Clustered,2,VC_0_0,2,0.2577,0.94993952,0.2448,1,1,1,1.0
1,Achromobacter~phage~JWAlpha,Caudovirales,Podoviridae,Jwalphavirus,5_0,Clustered,13,VC_5_0,13,0.5342,1.0,0.5342,5,1,1,0.8205
2,Achromobacter~phage~JWX,Caudovirales,Siphoviridae,Jwxvirus,0_0,Clustered,2,VC_0_0,2,0.2577,0.94993952,0.2448,1,1,1,1.0
3,Achromobacter~phage~phiAxp-3,Caudovirales,Podoviridae,Jwalphavirus,5_0,Clustered,13,VC_5_0,13,0.5342,1.0,0.5342,5,1,1,0.8205
4,Acidianus~bottle-shaped~virus,Unassigned,Ampullaviridae,Ampullavirus,15_0,Clustered,3,VC_15_0,3,1.0,1.0,1.0,2,1,1,1.0
5,Acidianus~bottle-shaped~virus~2,Unassigned,Ampullaviridae,Unassigned,15_0,Clustered,3,VC_15_0,3,1.0,1.0,1.0,2,1,1,1.0
6,Acidianus~bottle-shaped~virus~3,Unassigned,Ampullaviridae,Unassigned,15_0,Clustered,3,VC_15_0,3,1.0,1.0,1.0,2,1,1,1.0
7,Acidianus~filamentous~virus~3,Ligamenvirales,Lipothrixviridae,Betalipothrixvirus,16_0,Clustered,6,VC_16_0,6,0.961,0.99987658,0.960
9,1,1,1,1.0
8,Acidianus~filamentous~virus~6,Ligamenvirales,Lipothrixviridae,Betalipothrixvirus,16_0,Clustered,6,VC_16_0,6,0.961,0.99987658,0.960
9,1,1,1,1.0
[xinpeng@login1.swan vcontact2_output]$ tail genome_by_genome_overview.csv
31161,k141_55140||0_partial,Unassigned,Unassigned,Unassigned,,Outlier,,,,,,,,,
31162,k141_5585||0_partial,Unassigned,Unassigned,Unassigned,,Outlier,,,,,,,,,
31163,k141_5585||1_partial,Unassigned,Unassigned,Unassigned,,Singleton,,,,,,,,,
31165,k141_6136||full,Unassigned,Unassigned,Unassigned,,Outlier,,,,,,,,,
31166,k141_8064||0_partial,Unassigned,Unassigned,Unassigned,,Outlier,,,,,,,,,
31168,k141_8064||2_partial,Unassigned,Unassigned,Unassigned,,Singleton,,,,,,,,,
31169,k141_8244||full,Unassigned,Unassigned,Unassigned,,Singleton,,,,,,,,,
31170,k141_8910||0_partial,Unassigned,Unassigned,Unassigned,,Singleton,,,,,,,,,
31171,k141_9381||0_partial,Unassigned,Unassigned,Unassigned,,Outlier,,,,,,,,,
31172,uncultured~crAssphage,Unassigned,Unassigned,Unassigned,,Singleton,,,,,,,,,
```

genome_by_genome_overview.csv: Contains all the taxonomic information to *reference genomes*, as well as all the clustering information (initial VC (VC_22), refined VC (VC_22_1)), confidence metrics, and misc scores.
One important note is that the taxonomic information *is not included* for user sequences.

This means that each user will need to find their genome(s) of interest and check to see if reference genomes are located in the same VC. If the user genome is within the same VC subcluster as a reference genome, then there's a very high probability that the user genome is part of the same genus.

If the user genome is in the same VC *but not the same subcluster as a reference*, then it's highly likely the two genomes are related at roughly genus-subfamily level. If there are no reference genomes in the same VC or VC subcluster, then it's likely that they are not related at the genus level at all. That said (more below), it is possible they could be related at a higher taxonomic level (subfamily, family, order)

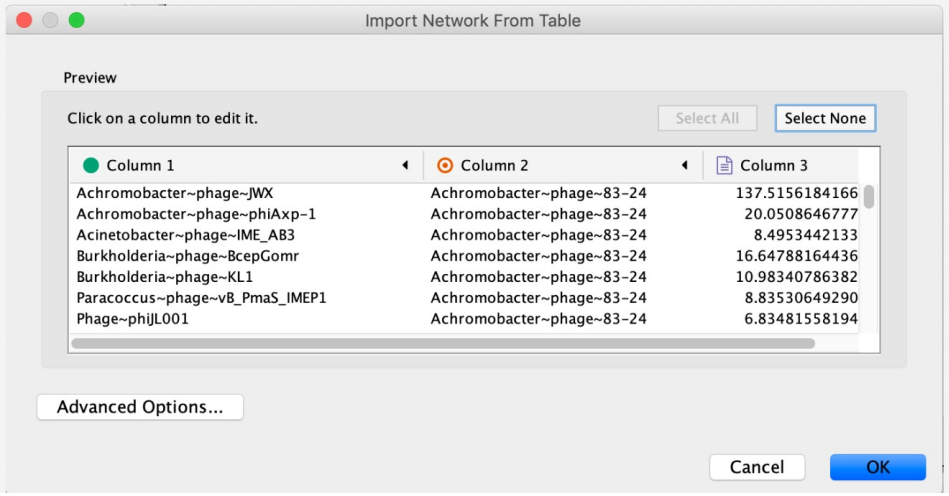


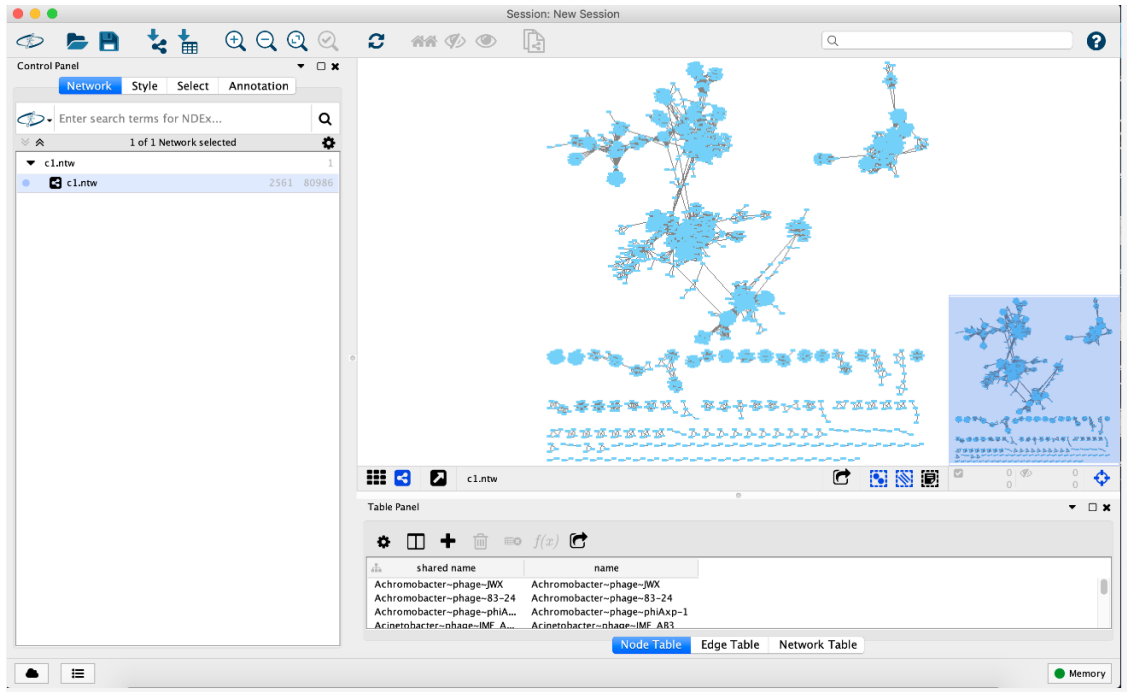
```
Achromobacter~phage~JWX Achromobacter~phage~83-24 147.64963409540033
Achromobacter~phage~phiAxp-1 Achromobacter~phage~83-24 20.06690087312625
Acinetobacter~phage~IME_AB3 Achromobacter~phage~83-24 7.707365136081943
Burkholderia~phage~BcepGomr Achromobacter~phage~83-24 16.50189198987472
Burkholderia~phage~KL1 Achromobacter~phage~83-24 12.936890467556296
Paracoccus~phage~vB_PmaS_IMEP1 Achromobacter~phage~83-24 8.047831169361991
Phage~phiJL001 Achromobacter~phage~83-24 6.043511214010593
Pseudomonas~phage~73 Achromobacter~phage~83-24 10.59348475339166
Pseudomonas~phage~M6 Achromobacter~phage~83-24 8.551976984660527
Pseudomonas~phage~MP1412 Achromobacter~phage~83-24 8.9571101194881
Pseudomonas~phage~PAE1 Achromobacter~phage~83-24 8.410358911739571
Pseudomonas~phage~PaMx28 Achromobacter~phage~83-24 14.037509256022034
Pseudomonas~phage~PaMx42 Achromobacter~phage~83-24 10.280678327675275
Pseudomonas~phage~PaMx74 Achromobacter~phage~83-24 13.968846663374716
Pseudomonas~phage~YuA Achromobacter~phage~83-24 8.9571101194881
Pseudomonas~phage~vB_Pae-Kakheti25 Achromobacter~phage~83-24 10.133259322439244
Pseudomonas~phage~vB_PaeS_SCH_Ab26 Achromobacter~phage~83-24 10.59348475339166
Rhodobacter~phage~RcTitan Achromobacter~phage~83-24 7.457322370714557
Roseobacter~phage~RDJL~Phi~1 Achromobacter~phage~83-24 23.334501565527436
Stenotrophomonas~phage~vB_SmaS-DLP_2 Achromobacter~phage~83-24 10.133259322439244
Achromobacter~phage~phiAxp-3 Achromobacter~phage~JWAlpha 176.0994765215513
Acinetobacter~phage~Presley Achromobacter~phage~JWAlpha 17.493785664679777
Alteromonas~phage~vB_AmaP_AD45-P1 Achromobacter~phage~JWAlpha 5.514220186578267
Delftia~phage~RG-2014 Achromobacter~phage~JWAlpha 63.114156500654175
Dinoroseobacter~phage~DFL12phi1 Achromobacter~phage~JWAlpha 46.43389646882473
Enterobacter~phage~EcP1 Achromobacter~phage~JWAlpha 31.511795644524472
```

Import it into the cytoscape

https://www.protocols.io/view/applying-vcontact-to-viral-sequences-and-visualizi-kqdg3pnql25z/v5

1. Click on 'Column 1' and under *Meaning*, select *Source Node* (little green button).
2. Click on 'Column 2' and under *Meaning*, select *Target Node* (red bullseye).
3. Click on 'Column 3' and under *Meaning*, select *Edge Attribute* (purple file).
4. Select 'Ok.' One this happens, it might take a while to load the network.





- After vCONTACT2, most phages are not assigned to a known taxonomy group.
- We'll further use BLASTn to search unclassified viral contigs against RefSeq viral genomes(.

RefSeq: NCBI Reference Sequence Database

A comprehensive, integrated, non-redundant, well-annotated set of reference sequences including genomic, transcript, and protein.

```
ml biodata
mkdir viral_refseq_db
cd viral_refseq_db
cp  refseq_viral_genomic.fasta .
makeblastdb -in refseq_viral_genomic.fasta -dbtype nucl -out refseq_viral_genomic_db
cd ..

#!/bin/bash
```

```
#!/bin/bash
#SBATCH --job-name=blastn
#SBATCH --ntasks-per-node=32
#SBATCH --time=168:00:00
#SBATCH --mem=100gb
#SBATCH --output=blastn.%J.out
#SBATCH --error=blastn.%J.err
#SBATCH --partition=vinlab,batch,guest

ml blast/2.13

blastn -query $1 -db viral_refseq_db/refseq_viral_genomic_db -out viral_vs_viral_blast_results.txt -outfmt 6
-max_target_seqs 10 -evalue 1e-3 -num_threads 32
```

- Then we have a python program to filter the results:

```
ml biopython

awk '{print $2}' blast_results.txt > ncbi_ids.txt
```

```
from Bio import Entrez

# set email
Entrez.email = "xzhang55@huskers.unl.edu"

# read NCBI ID
with open("ncbi_ids.txt") as file:
    ncbi_ids = file.read().splitlines()

unique_ids = set(ncbi_ids)

# get tax
for ncbi_id in unique_ids:
    handle = Entrez.efetch(db="nucleotide", id=ncbi_id, rettype="gb", retmode="xml")
    records = Entrez.read(handle)
    for record in records:
        # extract tax information
        if 'GBSeq_taxonomy' in record:
            taxonomy = record['GBSeq_taxonomy']
            print(f"{ncbi_id}: {taxonomy}")
        else:
            print(f"{ncbi_id}: Taxonomy information not found")
    handle.close()
```

- Save it as python program and run it:

  python your_program_name.py

- Map those results with your viral contig ID, and make a table to report the taxonomy.

```
[xinpeng@login1.swan tax_virus]$ python ncbi_tax.py
NC_012696.1: Viruses; Duplodnaviria; Heunggongvirae; Uroviricota; Caudoviricetes; Schitoviridae; Rhodovirinae; Aorunvirus; Aorunviru
s EE36phi1
NC_020489.1: Viruses; Duplodnaviria; Heunggongvirae; Uroviricota; Caudoviricetes
NC_012697.1: Viruses; Duplodnaviria; Heunggongvirae; Uroviricota; Caudoviricetes; Schitoviridae; Rhodovirinae; Aorunvirus; Aorunviru
s V12
NC_030909.1: Viruses; Duplodnaviria; Heunggongvirae; Uroviricota; Caudoviricetes
NC_027334.1: Viruses; Duplodnaviria; Heunggongvirae; Uroviricota; Caudoviricetes
```

- For host prediction, we'll continue to use BLASTn to search against refseq bacteria genomes

```
#!/bin/bash
#SBATCH --job-name=blastn
#SBATCH --ntasks-per-node=32
#SBATCH --time=168:00:00
#SBATCH --mem=100gb
#SBATCH --output=blastn.%J.out
#SBATCH --error=blastn.%J.err
#SBATCH --partition=vinlab,batch,guest


ml blast/2.13
ml biodata

blastn -query $1 -db $BLAST/ref_prok_rep_genomes -out viral_vs_viral_blast_results.txt -outfmt 6 -max_target_seqs 10
-evalue 1e-3 -num_threads 32 -task megablast
```

- And write a python to filter the result

```
# read the input file

blast_output_file = "viral_vs_viral_blast_results.txt"
host_matches_file = "host_matches.txt"

# set thresholds
min_length = 2500
min_identity = 90.0
min_coverage = 75.0

with open(blast_output_file, 'r') as file, open(host_matches_file, 'w') as outfile:
    for line in file:
        fields = line.strip().split()
        query_id, subject_id, identity, length = fields[0], fields[1], float(fields[2]), int(fields[3])
        alignment_length = int(fields[7]) - int(fields[6])


#check the condition
        if length >= min_length and identity >= min_identity and (alignment_length / length * 100) >= min_coverage:
            outfile.write(line)

print(f"Filtered results saved to {host_matches_file}")
```

- After that, you will get a viral_vs_viral_blast_results.txt result file.
- Then copy the previous python file and run it again to get the bacteria information:

```
[xinpeng@login1.swan tax_host]$ python ncbi_tax.py
NZ_SCNN01000019.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Phyllobacteriaceae; Mesorhizobium
NZ_HG326652.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Nitrobacteraceae; Afipia
NZ_JACXWY010000003.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Boseaceae; Bosea
NZ_JH584235.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Sphingomonadales; Sphingomonadaceae; Sphingomonas
NZ_SCNN01000023.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Phyllobacteriaceae; Mesorhizobium
NZ_CP062803.1: Bacteria; Pseudomonadota; Betaproteobacteria; Burkholderiales; Burkholderiaceae; Cupriavidus
NZ_CALSBS010000013.1: Bacteria; Pseudomonadota; Gammaproteobacteria; Enterobacterales; Enterobacteriaceae; Pseudocitrobacter
NZ_AQHN01000055.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Rhizobiaceae; Rhizobium/Agrobacterium group; Rhi
zobium
NZ_SCNN01000002.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Phyllobacteriaceae; Mesorhizobium
NZ_HG326654.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Nitrobacteraceae; Afipia
NZ_FNTJ01000001.1: Bacteria; Pseudomonadota; Gammaproteobacteria; Pseudomonadales; Pseudomonadaceae; Pseudomonas
NZ_VSSS01000065.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Nitrobacteraceae; Bradyrhizobium
NZ_MDEO01000032.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Phyllobacteriaceae; Mesorhizobium
NZ_SCNN01000004.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Phyllobacteriaceae; Mesorhizobium
NZ_CP013480.3: Bacteria; Pseudomonadota; Betaproteobacteria; Burkholderiales; Burkholderiaceae; Pandoraea
NZ_UFSI01000002.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Nitrobacteraceae; Afipia
NZ_JACWMY010000007.1: Bacteria; Bacteroidota; Sphingobacteriia; Sphingobacteriales; Sphingobacteriaceae; Mucilaginibacter
NZ_JACXWY010000016.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Boseaceae; Bosea
NZ_MDEO01000035.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Phyllobacteriaceae; Mesorhizobium
NZ_CP012268.1: Bacteria; Pseudomonadota; Gammaproteobacteria; Enterobacterales; Enterobacteriaceae; Cronobacter
NZ_CP012257.1: Bacteria; Pseudomonadota; Gammaproteobacteria; Enterobacterales; Enterobacteriaceae; Cronobacter
NZ_SCNN01000030.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Phyllobacteriaceae; Mesorhizobium
NZ_CABPSF010000004.1: Bacteria; Pseudomonadota; Betaproteobacteria; Burkholderiales; Burkholderiaceae; Pandoraea
NZ_BQHO01000039.1: Bacteria; Pseudomonadota; Gammaproteobacteria; Pseudomonadales; Pseudomonadaceae; Pseudomonas
NZ_JAJSPR010000017.1: Bacteria; Pseudomonadota; Gammaproteobacteria; Pseudomonadales; Pseudomonadaceae; Pseudomonas
NZ_CP044544.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Nitrobacteraceae; Bradyrhizobium
NZ_FNBZ01000005.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Boseaceae; Bosea
NZ_JACXWY010000001.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Boseaceae; Bosea
NZ_LR025742.1: Bacteria; Pseudomonadota; Betaproteobacteria; Burkholderiales; Burkholderiaceae; Burkholderia; Burkholderia cepacia c
omplex
NZ_CP044218.1: Bacteria; Pseudomonadota; Alphaproteobacteria; Hyphomicrobiales; Phyllobacteriaceae; Mesorhizobium
NZ_CP062804.1: Bacteria; Pseudomonadota; Betaproteobacteria; Burkholderiales; Burkholderiaceae; Cupriavidus
```

- check the phage taxonomy and host prediction if they match or not.
- viruses in the sample should infect the bacteria in the sample; could check if the predicted host of viruses and MAG taxonomy match or not.
- All of those chould be in your report.

Summary

- We can identify viral contigs from the total contigs assembled from the algal microbiome reads.
- We do the gene prediction, annotation, taxonomy and host prediction for the phages.
- Report the results of each step, try to combine the viral taxonomy, host taxonomy, and MAG taxonomy (make network or phylogeny tree).
- Try to find if the AMG could help the bacteria and algae.

Any questions?