metagenlab/diag$_{p}ipelines Documentation$

Release stable

May 19, 2022

Contents:

CHAPTER 1

Cookbook

Routine procedures for diagnostic purposes using microbial genomics and metagenomics.

Workflows for epidemiology, anti-microbial resistance genotyping and virulence factors identification have been implemented using the Snakemake workflow management system with bioconda integration for software dependency. Docker images of main releases are available.

1.1 Dependencies

1.1.1 Docker

Install the CE version following these instructions for ubuntu. Also make sure you have created the docker group and that you can run docker without sudo following these instruction. If you can't have access to the internet when inside a Docker container, apply those changes.

```
docker run hello-world
docker pull metagenlab/diag_pipelines:latest
docker run -t --rm metagenlab/diag_pipelines:latest sh -c "ping www.google.com"
```

Our Docker image is fit for a user called `pipeline_user` whose UID is `1080`. It is advised to create this user on your computer before using the Docker image to run your analysis.

```
sudo useradd -G docker,sudo -u 1080 pipeline_user
sudo mkdir /home/pipeline_user/
sudo chown pipeline_user -R /home/pipeline_user/
sudo passwd pipeline_user
```

Alternatively, you can run the Docker as root (`--user root`) but the created folders will belong to the root user of your computer.

1.2 General use

Once you have pulled the docker image on your computer:

```
docker run -t --rm \
--mount source="$(pwd)",target=/home/pipeline_user/data/analysis/,type=bind \
metagenlab/diag_pipelines:latest \
sh -c 'snakemake --snakefile $pipeline_folder/workflows/full_pipeline.rules \
--use-conda --conda-prefix $conda_folder --configfile config.yaml'
```

Update the config file for your needs. If you have read files you want to analyse, they should be stored in the `links` folder from your current working directory.

1.3 Run one of the 4 workflows

The pipeline implement four main workflows.

1. *Epidemiological analysis*
2. *Annotation of virulence factors*
3. *Annotation of resistance markers*
4. *Characterization of one or multiple strains*

An *html* report summarizing results is generated upon completion of the workflow.

1.3.1 Epidemiological analysis

```
docker run -t --rm \
--mount source="$(pwd)",target=/home/pipeline_user/data/analysis/,type=bind \
metagenlab/diag_pipelines:latest \
sh -c 'snakemake --snakefile $pipeline_folder/workflows/full_pipeline.rules\
--use-conda --conda-prefix $conda_folder --configfile config.yaml\
epidemiology'
```

This will perform quality checks, map reads against one or multiple reference genome, calculate pairwise number of SNPs and generate a minimum spanning tree. The reference genome can be one of the assembled genome or an assembly available on the NCBI website. The analysis can also be restricted to the core genome as defined by existing cgMLST schemes or by computing a custom core genome with help of parsnp (see documentation https://metagenlabdiag-pipelines.readthedocs.io/en/latest/core_genomes.html).

1.3.2 Annotation of virulence factors

```
docker run -t --rm \
--mount source="$(pwd)",target=/home/pipeline_user/data/analysis/,type=bind \
metagenlab/diag_pipelines:latest \
sh -c 'snakemake --snakefile $pipeline_folder/workflows/full_pipeline.rules\
--use-conda --conda-prefix $conda_folder --configfile config.yaml\
virulence'
```

This will perform quality checks, assemble the genome and search for known virulence factors from the VFDB database.

1.3.3 Annotation of resistance markers

```
docker run -t --rm \
--mount source="$(pwd)",target=/home/pipeline_user/data/analysis/,type=bind \
metagenlab/diag_pipelines:latest \
sh -c 'snakemake --snakefile $pipeline_folder/workflows/full_pipeline.rules\
--use-conda --conda-prefix $conda_folder --configfile config.yaml\
resistance'
```

This will perform quality checks, assemble the genome and search for known antibiotic resistance determinants with help of the rgi software and CARD database.

1.3.4 Characterization of one or multiple strains

```
docker run -t --rm \
--mount source="$(pwd)",target=/home/pipeline_user/data/analysis/,type=bind \
metagenlab/diag_pipelines:latest \
sh -c 'snakemake --snakefile $pipeline_folder/workflows/full_pipeline.rules\
--use-conda --conda-prefix $conda_folder --configfile config.yaml\
strain_characterization'
```

This will perform quality checks, assemble the genome and search for known antibiotic resistance determinants with help of the rgi software and CARD database and search for known virulence factors from the VFDB database.

1.4 Generating specific files of interest

If you want to execute a specific analysis, you can request files of interest for a particular analysis. Consult the full documentation to know what files can be generated (http://metagenlabdiag-pipelines.readthedocs.io/en/latest/). Main examples are provided below:

```
docker run -t --rm \
--mount source="$(pwd)",target=/home/pipeline_user/data/analysis/,type=bind \
metagenlab/diag_pipelines:latest \
sh -c 'snakemake --snakefile $pipeline_folder/workflows/full_pipeline.rules\
--use-conda --conda-prefix $conda_folder --configfile config.yaml\
report/multiqc_assembly/multiqc_report.html'
```

This will assemble and annotate every samples, and generate a multiqc report for all samples.

```
docker run -t --rm \
--mount source="$(pwd)",target=/home/pipeline_user/data/analysis/,type=bind \
metagenlab/diag_pipelines:latest \
sh -c 'snakemake --snakefile $pipeline_folder/workflows/virulence.rules\
--use-conda --conda-prefix $conda_folder --configfile config.yaml\
virulence_summary.xlsx'
```

This will generate a summary excel file for the virulence factors of the samples, based on the virulence factors annotated in the file defined on the config file.

```
docker run -t --rm \
--mount source="$(pwd)",target=/home/pipeline_user/data/analysis/,type=bind \
metagenlab/diag_pipelines:latest \
sh -c 'snakemake --snakefile $pipeline_folder/workflows/typing.rules\
```

(continues on next page)

(continued from previous page)

```
--use-conda --conda-prefix $conda_folder --configfile config.yaml\
typing/freebayes_joint_genotyping/cgMLST/bwa/distances_in_snp.xlsx'
```

This will generate a snp-distance matrix of all samples, only on the core genome defined by ridom of the species defined in the *species* variable of the config file, mapped with bwa on the reference genome used by ridom (which is *Staphylococcus aureus* COL substrain, *id* 33148 from the NCBI Assembly database).

```
docker run -t --rm \
--mount source="$(pwd)",target=/home/pipeline_user/data/analysis/,type=bind \
metagenlab/diag_pipelines:latest \
sh -c 'snakemake --snakefile $pipeline_folder/workflows/resistance.rules\
--use-conda --conda-prefix $conda_folder --configfile config.yaml\
report/typing/mlst/summary.xlsx'
```

This will generate an Excel summary file of the MLST of all samples, based on the software mlst.

1.4.1 All Deliverables

Here is a list of all deliverables currently available:

CHAPTER 2

Generalities

2.1 Defining variables

As a general rules, any `variable` referenced in this documentation must be either:

- Defined in the yaml config file that is passed to snakemake by `--configfile`
- Defined directly in the snakemake command by `--config variable=$value`

2.1.1 General variables

- Pass `--cores number_of_cores` to the snakemake command to define the number of cores you want to use for your analysis

An example of each variable value is provided in `config.yaml` of the github repository.

2.2 Configuration file

Example of configuration file. All parameters are not necessary for all 4 workflows.

```
#mandatory, one of them or both
sra_samples: example_sra_samples.tsv
local_samples: example_local_samples.tsv
#assembly
minimum_quality_base: 28
minimum_read_length: 50
sliding_window_size: 5
sliding_window_quality_threshold: 20
adapter_file_name: NexteraPE-PE.fa
adapter_removal_param1: 3
adapter_removal_param2: 25
adapter_removal_param3: 6
```

(continues on next page)

(continued from previous page)

```
cov_cutoff: 5
spades_kmer_sizes: 21,33,55,77,99,111,127
#resistance and typing
species: Mycobacterium_tuberculosis
#typing
minimum_coverage_for_calling: 10
minimum_alternate_fraction_for_calling: 0.75
snp_threshold: 0
minimum_spanning_tree_size: 10
phylogeny_image_size: 800
# reference for snp calling:
# 1. cgMLST
# 2. one or multiple samples listed in local_samples/sra_samples (assembled genomes)
# 3. one genome from NCBI using assembly UID (eg: 33148 for S. aureus COL genome, see␣
↪page https://www.ncbi.nlm.nih.gov/assembly/GCF_000012045.1/)
# 4. combination of options 1-3 (comma separated list, no spaces: "cgMLST,ecoli39,
↪33148")
reference: "cgMLST"

# snp calling method: either "freebayes_joint_genotyping" or "gatk_gvcfs"
snp_caller: "gatk_gvcfs"

# mapping: either bwa or bwa_stringent
mapping: "bwa"

#virulence
virulence_percentage_identity_cutoff: 80
virulence_coverage_cutoff: 70
```

2.3 Logging functions

Logging processes are defined in the file `workflows/logging.rules`.

2.3.1 Parameters

The variable `logging_folder` can be defined in the `config.yaml` or passed to snakemake with `--config`. If it is not defined, the default value will be the folder `logs/`.

2.3.2 Saved files

Each time an effective snakemake run is started, a folder named with the current UTC datetime is created. A variable number of files will be copied there, so that replication of the run is possible:

- The snakefile passed to snakemake
- The config file
- The full command used, copied into the file `cmd.txt`
- The parameter files defining the SRA and the local samples, if they exist

The logs of every command run during the execution of the workflow will then be stored in this folder.

2.4 Determining sample names

Sample naming and matching to fastq files are handled in the file `workflows/making_sample_dataset.rules`.

2.4.1 Parameters

- The variable `link_directory` can be defined. Its default value is `links/`. Read data for local samples will be searched in this directory.
- *Local samples* can be defined based on a tabulated file whose full path can be passed to the variable `local_samples`
- *SRA samples* can be defined based on the tabulated file whose full path can be passed to the variable `sra_samples`

2.4.2 Local samples

The tabulated file whose path is defined by `local_samples` must contain at least two columns: *SampleName* and *ScientificName*.

Table 1: Local data example

SampleName	ScientificName
S10	Staphylococcus aureus
S1	Staphylococcus aureus

For each entry, there must be in the folder defined by the `link_directory` variable, two files (for paired reads) or only one (for single reads) whose filename starts by one and only one entry of the *SampleName* columns. For instance, the files `S10_001_R1_L001.fastq.gz` and `S10_001_R2_L001.fastq.gz` in the folder defined by the `link_directory` variable will be matched to the sample name `S10`. The matching is performed by using regular expressions to end the search at non alphanumeric characters or by the end of the word, thus the sample name `S1` will actually not match `S10_001_R1_L001.fastq.gz` nor `S10_001_R2_L001.fastq.gz`.

If needed, an *OldSampleName* column can be added to the file, when the read filenames and the desired new sample names can not be matched simply by testing the identity at the start of both names.

Table 2: Local data example with old sample names

SampleName	ScientificName	OldSampleName
S10	Staphylococcus aureus	Staaur-10
S1	Staphylococcus aureus	Staaur-1

In this case, the files `Staaur-10_S10_L001_R1_001.fastq.gz` and `Staaur-10_S10_L001_R2_001.fastq.gz` in the folder defined in `link_directory` will be matched to the sample name `S10`. Similarly, `Staaur-1` will actually not match `Staaur-10_S10_L001_R1_001.fastq.gz`.

2.4.3 SRA samples

The tabulated file whose path is defined by `sra_samples` can be the RunInfo files that are downloadable directly through the SRA NCBI and can be passed without any modification. If the variable `use_library_name` with any

value is passed during execution, the column *LibraryName* will be used for naming the samples, instead of *SampleName* (which can be useful for badly formatted Sra Run Info files). If you want to format your own SRA samples without a RunInfo file from the NCBI, 4 columns must be defined:

Table 3: SRA data example

Run	SampleName	LibraryLayout	ScientificName
ERR2130394	SE1	single	Mycobacterium tuberculosis
SRR6936587	XTB-14-012	single	Mycobacterium tuberculosis

2.5 Workflows and Rules

Current available workflows are implemented in the folder `workflows`. Each workflow will depend on `rules`, stored in the folder of the same name, and can also depend on other workflows. `rules` are sorted with respect to their general function in different folders.

2.5.1 Rules

Table 4: All rules in productions

Location	Rule name
annotation/prokka.rules	annotate_with_prokka
annotation/prokka.rules	annotate_with_prokka_unfiltered_assembly
annotation/prokka.rules	create_blast_database_from_protein_sequences
annotation/prokka.rules	create_blast_database_from_contig_sequences
annotation/prokka.rules	remove_fasta_part_from_gff
annotation/virulence.rules	blast_virulence_protein_to_proteome_or_contigs
annotation/virulence.rules	remove_redundancy_from_blast_results
annotation/virulence.rules	extract_protein_sequences_from_blast_results
annotation/virulence.rules	add_description_to_blast_results

Continued on next page

Table 4 – continued from previous page

Location	Rule name
annotation/virulence.rules	merge_samples_summary
assembly/spades.rules	correct_error_paired_reads_with_spades
assembly/spades.rules	correct_error_single_reads_with_spades
assembly/spades.rules	assemble_genome_paired_reads_with_spades
assembly/spades.rules	assemble_genome_single_reads_with_spades
core_genome/bed_creation.rules	extract_core_genome_parsnp_from_ref
downloading/fetch_references.rules	get_refseq_urls_complete_genomes
downloading/fetch_references.rules	download_all_complete_genomes_fasta
downloading/fetch_single_reference.rules	download_reference_from_refseq
downloading/fetch_single_reference.rules	download_reference_from_nucleotide
downloading/fetch_single_reference.rules	get_strain_subvalue_identifier_reference
downloading/fetch_single_reference.rules	download_gff_for_reference_from_refseq
downloading/fetch_virulence_factors.rules	fetch_virulence_factors_from_uniprot_accessions
downloading/linking_references_for_core_genome_schemes.rules	link_reference_genome_for_cgMLST
downloading/linking_references_for_core_genome_schemes.rules	link_reference_genome_for_parsnp
downloading/linking_references_for_core_genome_schemes.rules	link_full_genome

Continued on next page

Table 4 – continued from previous page

Location	Rule name
genotyping/freebayes_first_pass.rules	genotype_with_freebayes_one_sample
genotyping/freebayes.rules	genotype_with_freebayes_for_resistance
genotyping/freebayes.rules	genotype_with_freebayes_on_all_samples
genotyping/freebayes.rules	genotype_with_freebayes_one_sample_second_pass
genotyping/gatk.rules	create_dictionary_for_reference
genotyping/gatk.rules	genotype_with_HaplotypeCaller_GATK_BP_RESOLUTION
genotyping/gatk.rules	merge_gvcf_files_with_GenomicsDBImport_GATK
genotyping/gatk.rules	merge_gvcf_files_with_CombineGVCFs_GATK
genotyping/gatk.rules	genotype_with_GenotypeGVCFs_GATK
mapping/bwa.rules	map_paired_reads_with_bwa
mapping/bwa.rules	map_single_reads_with_bwa
mapping/bwa.rules	filter_reads_on_quality
mapping/bwa.rules	remove_duplicates_from_mapping
mapping/find_closest_genomes.rules	sketch_kmers_complete_genomes_with_mash
mapping/find_closest_genomes.rules	calculate_distance_to_complete_genomes_with_mash
mapping/find_closest_genomes.rules	extract_closest_genomes_to_all_samples

Continued on next page

Table 4 – continued from previous page

Location	Rule name
mapping/indexing_files.rules	index_reference_fasta
mapping/indexing_files.rules	index_bam_file
phylogeny/image_creation.rules	convert_phylogeny_to_image_with_st
phylogeny/image_creation.rules	convert_phylogeny_to_image_no_st
phylogeny/raxml.rules	compute_phylogeny_with_raxml
phylogeny/raxml.rules	compute_phylogeny_bootstraps_with_raxml
quality/assembly_filtering.rules	copy_raw_assembly_to_reference_folder
quality/assembly_filtering.rules	extract_contig_coverage
quality/assembly_filtering.rules	filter_contigs_on_coverage
quality/assembly_filtering.rules	extract_contigs_longer_than_500bp
quality/assembly_filtering.rules	rename_contigs
quality/contamination.rules	calculate_distance_paired_reads_from_refseq_genomes_with_mash
quality/contamination.rules	calculate_distance_single_reads_from_refseq_genomes_with_mash
quality/contamination.rules	get_taxonomy_from_mash_results
quality/contamination.rules	format_distances_from_mash_results
quality/contamination.rules	format_tsv_to_xlsx_mash_results

Continued on next page

Table 4 – continued from previous page

Location	Rule name
quality/contamination.rules	merge_all_xlsx_mash_results
quality/trimmomatic.rules	trim_paired_reads_with_trimmomatic
quality/trimmomatic.rules	trim_single_reads_with_trimmomatic
read_manipulation/get_reads.rules	copy_fastq_paired_from_link
read_manipulation/get_reads.rules	copy_fastq_single_from_link
read_manipulation/get_sras.rules	download_sra_single
read_manipulation/get_sras.rules	download_sra_paired
report_generation/fastqc.rules	assess_quality_single_reads_with_fastqc
report_generation/fastqc.rules	assess_quality_paired_reads_with_fastqc
report_generation/fastqc.rules	unzip_fastqc_single
report_generation/fastqc.rules	unzip_fastqc_paired
report_generation/multiqc.rules	create_multiqc_report_for_assembly
report_generation/multiqc.rules	create_multiqc_report_for_mapping
report_generation/prepare_files_for_multiqc.rules	copy_result_files_mapping_paired
report_generation/prepare_files_for_multiqc.rules	copy_result_files_mapping_single
report_generation/prepare_files_for_multiqc.rules	copy_result_files_assembly

Continued on next page

Table 4 – continued from previous page

Location	Rule name
report_generation/qualimap.rules	assess_mapping_with_qualimap
report_generation/quast.rules	calculate_assembly_statistics_with_quast
typing/mlst.rules	determine_mlst
typing/mlst.rules	merge_mlst_from_all_samples
typing/mlst.rules	determine_mlst_reference_genome
typing/mlst.rules	generate_xlsx_file_from_mlst_results
typing/snp_distance.rules	distance_columns_to_matrix
typing/snp_distance.rules	compute_minimum_spanning_tree_with_st
typing/snp_distance.rules	compute_minimum_spanning_tree_no_st
vcf_manipulation/calculate_differences.rules	calculate_pairwise_distances_by_type
vcf_manipulation/calculate_differences.rules	get_pairwise_snps_positions_by_type
vcf_manipulation/calculate_differences.rules	calculate_distance_with_ref_by_type
vcf_manipulation/calculate_differences.rules	agregate_distances_from_joint_genotyping_by_type
vcf_manipulation/create_alignment_for_phylogeny.rules	merge_multiallelic_by_sample
vcf_manipulation/create_alignment_for_phylogeny.rules	extract_alternative_positions_and_unknown_positions
vcf_manipulation/create_alignment_for_phylogeny.rules	create_consensus_sequence_by_sample

Continued on next page

Table 4 – continued from previous page

Location	Rule name
vcf_manipulation/create_alignment_for_phylogeny.rules	concatenate_consensus_fasta_files_all_samples
vcf_manipulation/extract_cgMLST.rules	extract_cgMLST_regions_from_vcf
vcf_manipulation/filtering.rules	decompose_multiallelics_and_normalize
vcf_manipulation/filtering.rules	filter_on_coverage
vcf_manipulation/filtering.rules	filter_on_frequency_per_sample
vcf_manipulation/filtering.rules	extract_allele_by_type_from_gatk_gvcfs
vcf_manipulation/filtering.rules	extract_allele_by_type_from_freebayes_joint_genotyping
vcf_manipulation/filtering.rules	extract_core_genome_parsnp
vcf_manipulation/indexing.rules	compress_vcf
vcf_manipulation/indexing.rules	index_vcf
vcf_manipulation/indexing.rules	sort_vcf
vcf_manipulation/splitting_merging.rules	extract_sample_entry_from_vcf
vcf_manipulation/splitting_merging.rules	merge_all_samples_entries_into_vcf
vcf_manipulation/splitting_merging.rules	merge_all_vcf_freebayes_first_pass
vcf_manipulation/splitting_merging.rules	merge_freebayes_second_pass
annotation/resistance/format_xlsx.rules	convert_tsv_to_xlsx

Continued on next page

Table 4 – continued from previous page

Location	Rule name
annotation/resistance/format_xlsx.rules	merge_rgi_or_mykrobe_xlsx
annotation/resistance/m_tuberculosis.rules	create_reference_lists_from_databases
annotation/resistance/m_tuberculosis.rules	merge_nucleotides_and_codons_bed_files
annotation/resistance/m_tuberculosis.rules	extract_all_locus_tags
annotation/resistance/m_tuberculosis.rules	fetch_locus_tag_sequences_from_accession
annotation/resistance/m_tuberculosis.rules	remove_shift_from_fasta_sequences
annotation/resistance/m_tuberculosis.rules	shift_positions_from_genotype_vcf
annotation/resistance/m_tuberculosis.rules	apply_genotype_to_fasta
annotation/resistance/m_tuberculosis.rules	extract_mutated_positions
annotation/resistance/m_tuberculosis.rules	extract_reference_positions
annotation/resistance/m_tuberculosis.rules	format_resistance_results
annotation/resistance/m_tuberculosis.rules	add_translation_to_mutated_codons
annotation/resistance/m_tuberculosis.rules	format_mutated_nucleotides
annotation/resistance/m_tuberculosis.rules	merge_mutated_nucleotides_and_codons
annotation/resistance/m_tuberculosis.rules	merge_non_empty_results
annotation/resistance/mykrobe.rules	search_resistance_paired_reads_with_mykrobe

Continued on next page

Table 4 – continued from previous page

Location	Rule name
annotation/resistance/mykrobe.rules	search_resistance_single_reads_with_mykrobe
annotation/resistance/mykrobe.rules	generate_mykrobe_tsv_file_from_json_file
annotation/resistance/rgi.rules	search_resistance_with_rgi
annotation/resistance/rgi.rules	extract_resistance_from_ontology
annotation/resistance/rgi.rules	generate_rgi_tsv_file_from_json_file
annotation/resistance/summarize_results.rules	summary_csv_excel_file
annotation/resistance/summarize_results.rules	write_congruent_results_fasta
annotation/resistance/summarize_results.rules	merge_summary_xlsx_files

CHAPTER 3

Core genome determination

Core genomes can be calculated by three different means. Core genomes definition from ridom and enterobase are already included in the Docker image, in the folder `/home/pipeline_user/core_genomes/cgMLST`, and the references they use in `/home/pipeline_user/references/`.

3.1 Ridom

cgMLST scheme from ridom can be extracted directly for theses species

Table 1: Available cgMLST schemes from ridom

Species	Taxonomy ID	Ridom ID	Reference genome assembly ID
Acinetobacter_baumannii	470	3956907	39528
Enterococcus_faecium	1352	991893	526908
Klebsiella_pneumoniae	573	2187931	31388
Listeria_monocytogenes	1639	690488	264498
Legionella_pneumophila	446	1025099	30068
Mycobacterium_tuberculosis	1773	741110	538048
Staphylococcus_aureus	1280	141106	33148
Clostridioides_difficile	1496	3560802	6374038

A bed file is constructed from the locus target file, using coordinates from the start and length columns of the csv file file available on the ridom website.

3.1.1 Example

```
snakemake --snakefile $pipeline_folder/workflows/core_genome/make_ridom.rules \
core_genomes/cgMLST/Staphylococcus_aureus.bed \
--use-conda --conda-prefix $conda_folder
```

will create a BED file in `core_genomes/cgMLST/Staphylococcus_aureus.bed` which defines the core genomic regions in the genome of the assembly ID `33148` (*Staphylococcus aureus* COL).

3.2 Enterobase

cgMLST scheme from enterobase is available for:

Table 2: Available cgMLST schemes from enterobase

Species	Taxonomy ID	Enterobase ID	Reference genome assembly ID	Scheme
Escherichia_coli	562	ESCwgMLST	79781	cgMLSTv1
Salmonella_enterica	28901	SALwgMLST	359488	cgMLSTv1

A bed file for each reference genome, based on the locus tags present in this genome, is constructed. For instance, over the 3002 loci of the *Salmonella* cgMLSTv1, 69 come from a different genome than the reference `359488`. For *E. coli*, only 15 loci are missing for the reference assembly (`79781`), out of 2498.

3.2.1 Example

```
snakemake --snakefile $pipeline_folder/workflows/core_genome/make_enterobase.rules \
core_genomes/cgMLST/Salmonella_enterica.bed \
--use-conda --conda-prefix $conda_folder
```

will create a BED file in `core_genomes/cgMLST/Salmonella_enterica.bed` defining the core genomic regions in the genome of the assembly ID `359488` (*Salmonella enterica* subsp. enterica serovar Typhimurium str. D23580).

3.3 ParSNP

For species unavailable on either resource, core genome can be calculated using parsnp and the complete genomes of the species available on RefSeq. As ParSNP is not available on bioconda, the binary must be downloaded from the ParSNP website and placed in your $PATH.

3.3.1 Example

```
snakemake --snakefile $pipeline_folder/workflows/core_genomes/make_parsnp.rules \
core_genome/parsnp/Morganella_morganii/parsnp.xmfa \
--use-conda --conda-prefix $conda_folder
```

will calculate the core genome with parSNP with every complete genome of *Morganella morganii* available in RefSeq.

If you wish to create a new parSNP core genome definition with the Docker image (that include the `parsnp` binary), do not link any `references` or `core_genomes` from your working directory.

CHAPTER 4

Assembly and quality

Aggregates rules for assembling genomes and performing various quality control checks.

4.1 Parameters

- `cov_cutoff`: contigs whose coverage is below this cutoff will be excluded from the final assembly
- `adapter_file_name`: look for the adaptor for this library preparation kit (possible values)
- `adapter_removal_param1`, `adapter_removal_param2`, `adapter_removal_param3`: parameters for adapter trimming (reference)
- `minimum_quality_base`: leading and trailing bases below this quality will be removed
- `minimum_read_length`: reads shorter than this threshold after trimming will be discarded (be careful when using reads from SRA!)

4.2 Deliverables

4.2.1 Assembly quality

- `report/multiqc_assembly/multiqc_report.html`: quality control report based on the results of **fastqc**, **trimmomatic**, **qualimap**, **quast** and **prokka** for every sample
- `report/contamination/low_coverage_contigs/{sample}.html`: quality control report based on the results of **fastqc**, **trimmomatic**, **qualimap**, **quast** and **prokka** for every sample
- `report/multiqc_assembly/multiqc_report.html`: quality control report based on the results of **fastqc**, **trimmomatic**, **qualimap**, **quast** and **prokka** for every sample

4.2.2 Assembly and annotation

- `samples/{sample}/assembly/spades/`: folder containing all files from **SPADES** assembly (contigs.fasta, assembly_graph.fastg, contigs.paths,...)
- `samples/{sample_name}/annotation/`: folder containing all annotation files from **prokka**

CHAPTER 5

Virulence

Depends on the *Assembly and quality* workflow.

Fig. 1: Steps to identify virulence factors

After the annotation of the genome, virulence factors are search by two different ways:

- Factors longer than 50 amino acids are search by `blastp` over the proteome annotated by prokka
- Factors shorter than 50 amino acids are search by `tblastn` directly over the contigs of the assembly

5.1 Parameters

- `virulence_factors`: file with list of uniprot accession of virulence factors. An example is available in the folder `data/staph/db/`

Table 1: Virulence data example

gene	uniprot_accession	description
hla	P09616	Alpha-hemolysin
hld	P0C1V1	Delta-hemolysin
hlgA	P0A074	Gamma-hemolysin component A
hlgB	P0A077	Gamma-hemolysin component B

- `virulence_percentage_identity_cutoff`: amino acid identity cut off for considering a match
- `virulence_coverage_cutoff`: coverage cut off for considering a match

5.2 Deliverables

- `virulence_summary.xlsx`: summary of virulence proteins found in every samples (one per sheet).

CHAPTER 6

Resistance

Depends on the *Epidemiology* workflow (for genotyping samples)

6.1 Parameters

- `species`: Species name to determine which software are available to run for your sample

6.2 Available softwares

6.2.1 Comprehensive Antibiotic Resistance Database (CARD)

The Resistance Gene Identifier (RGI) from the Comprehensive Antibiotic Resistance Database, version `3.2.1` is used, with data version `1.1.9`. Predictions are based on the assembled contigs.

Ontology

The CARD ontology is parsed after the analysis are run to summarize the antibiotics for which a gene conferring resistance has been identified (testing for the relationship equals to `confers_resistance_to_drug` or `confers_resistance_to`). Find the result of the parsing for each sample in:

- `samples/{sample}/resistance/rgi_ontology.xlsx`

6.2.2 Mykrobe

Mykrobe can be used on *Staphylococcus aureus* and *Mycobacterium tuberculosis* samples only. Predictions are based directly on the fastq reads. Version `0.5.6` is used.

Parameters

For *M. tuberculosis*, two different panels of mutations can be analysed, by defining the variable `mykrobe_panel`. Two different values are possible:

- `bradley-2015`, from Bradley et al. 2015, Nature Communications (default value)
- `walker-2015`, from Walker et al. 2015, Lancet Infectious Diseases

The threshold of confidence for Mykrobe can also be defined with `mykrobe_confidence`. The default value is `10`.

6.3 *Mycobacterium tuberculosis* specific analyses

If the value of `species` is `Mycobacterium_tuberculosis`, specific variant markers of resistance can be searched by mapping to the genome of H37Rv, genotyping with GATK and annotating the resulting VCF. Different sources of annotation can be used to search markers:

- Walker et al. 2015 Lancet Infectious Diseases
- CARD database
- Miotto et al. 2017 European Respiratory Journal
- Bradley et al. 2015 Nature Communications

6.4 Deliverables

6.4.1 rgi and mykrobe analysis

- `report/resistance/rgi_summary.xlsx`: summary of RGI results for every samples (one per sheet)
- `report/resistance/mykrobe_summary.xlsx`: summary of Mykrobe results for every samples (one per sheet, only for *Mycobacterium* genus or *Staphylococcus aureus*)
- `report/resistance/{sample}_rgi_report.html`: summary of RGI results as html document with cross references to the CARD database

6.4.2 *Mycobacterium tuberculosis* SNPs variant associated to resistance

- `sample/{sample_name}/resistance/bwa/miotto_high_moderate_minimum_confidence_annotated/mutations.vcf`: VCF files of all markers in Miotto et al. 2017 European Respiratory Journal
- `sample/{sample_name}/resistance/bwa/mykrobe_annotated/mutations.vcf`: VCF files of all markers in Bradley et al. 2015 Nature Communications
- `sample/{sample_name}/resistance/bwa/walker_resistant_annotated/mutations.vcf`: VCF files of all markers in Walker et al. 2015 Lancet Infectious Diseases
- `sample/{sample_name}/resistance/bwa/rgi_annotated_full_2_0_0/mutations.vcf`: VCF files of all markers in version 2.0.0 of the CARD database

CHAPTER 7

Epidemiology

Depends on the *Assembly and quality* workflow (for determining the Sequence Types). The genotyping results depend on the quality assessment performed on the mapping to the reference genomes, thus each time genotyping is performed, a Multiqc report is available in `quality/multiqc/mapping_to_{ref}/multiqc_report.html` and the contamination results in `contamination/distances_formated.xlsx` for each sample.

7.1 Parameters

- `minimum_coverage_for_calling`: minimum of coverage for considering a genomic position when counting differences between samples. Any position (whether the genotype is identical to the reference, ie *GT=0* in the vcf, or different, ie *GT=1*) having a lower coverage will be masked.
- `minimum_alternate_fraction_for_calling`: minimum ratio of observations favouring an alternative allele over observations favouring the reference allele. Any position (*GT=0* or *GT=1*) not meeting this criteria will also be masked.
- `snp_threshold`: pairs of samples having less than this number of SNP differences will be linked in the final minimum spanning tree
- `minimum_spanning_tree_size`: size of the minimum spanning tree image, default is `10`
- `phylogeny_image_size`: size of the phylogeny image, default is `800`
- `species`

7.2 Available Genotypers

Genotyping can be performed with two different softwares:

7.2.1 Freebayes

Genotyping with Freebayes is done at the same time for all samples. This will enable us to keep coverage informations that we will need during the filtering steps.

Fig. 1: Simplified Directed Acyclic Graph (DAG) for Freebayes SNP calling and distance calculation

7.2.2 GATK

Genotyping with GATK is done in two pass. First, `HaplotypeCaller` is called on every sample using the option `--ERC BP_RESOLUTION`. The resulting gVCF file then contains the SNP calls plus the coverage at every position of the reference genome. Once every sample has been called with `HaplotypeCaller`, the gVCFs are merged (with `CombineGVCFs`) and the final vcf file is obtained with `GenotypeGVCFs`. Using `--ERC BP_RESOLUTION` enables us to keep the coverage information for each sample, at positions where SNPs were called in other samples. This information will be needed at the filtering step.

Fig. 2: Simplified Directed Acyclic Graph (DAG) for GATK (HaplotypeCaller and GenotypeGVCFs) SNP calling and distance calculation

7.3 Filtering

Fig. 3: Overview of the filtering process

7.3.1 Coverage filtering

All positions in every sample are filtered simultaneously, whether the genotype if REF or ALT in VCF. Any position will be masked (genotype is set to unknown, `GT=.`) if the coverage is lower than `minimum_coverage_for_calling`.

7.3.2 Frequency filtering

Filtering over the frequency of observation at each position is processed separately for each sample. Multiallelic entries from the VCF are splitted and normalized with vt. Then, for alternative genotype entries, the genotype is set to unknown if the ratio of observation of alternate base(s) (`FORMAT/AD[0:1]` in the VCF) over the total coverage of the position (`FORMAT/DP`) is lower than `minimum_alternate_fraction_for_calling`. Similarly, for reference genotype entries, the genotype is set to unknown if the ratio of the number of observations of reference base(s) (`FORMAT/AD[0:0]`) over the total coverage of the position (`FORMAT/DP`) is lower than `minimum_alternate_fraction_for_calling`. After the filtering is done, all samples are merged back together in the same VCF.

7.3.3 Type filtering

Entries in the VCF are filtered according to their type. We preferentially keep only entries defined as single nucleotide polymorphisms, but `indel` or `bnd` can also be extracted from the genotyping of freebayes, or `indel` or `mixed` from the genotyping of GATK.

7.3.4 Core genome filtering

Core genome filtering is performed at the end. Using any bed file defined using the mapping reference, only the positions in the filtered final VCF overlapping the bed regions are kept. See *Core genome determination* for the bed file definitions.

7.4 Calculating differences

Fig. 4: Calculating differences between pairs of samples and samples and reference

After the filtering has been performed, differences in snps are calculated between samples and against the reference. Every comparison is pairwise. At each position, if any of the two genotype is unknown because of the filtering (`GT="."`), this position will not be counted as a difference. Once every distances have been computed, they are agregated in a single file, transformed to a matrix and finally to a minimum spanning tree image with the `igraph` R package.

7.5 Phylogeny

Fig. 5: Steps to create phylogeny from the filtered genotype positions

After the filtering has been performed, the SNPs of each samples are replaced in the reference fasta file. However, the unknown positions are replace with `N`. Every sequence is then concatenated and a simple phylogeny is calculated with RAxML, without partitions, with the GTRCAT model.

7.6 Deliverables

The wildcard `{snp_caller}` in the following result files can have two values: `freebayes_joint_genotyping` or `gatk_gvcfs`.

- `typing/{snp_caller}/cgMLST/bwa/distances_in_snp_mst_no_st.svg`: minimum spanning tree of the distance in snps between every sample over the core genome as defined by ridom or enterobase. Available species and values for reference genomes are listed in the files in `data/core_genome_dbs/`.
- If the species under consideration has a multiple locus sequence type available, `typing/{snp_caller}/cgMLST/bwa/distances_in_snp_mst_with_st.svg` can be generated with each sample colored by its Sequence Type.
- `phylogeny/{snp_caller}/cgMLST/bwa/phylogeny_no_st.svg`: a phylogeny based on the alignments of the core SNPs, using RAxML. Available species and values for reference genomes are listed in the files in `data/core_genome_dbs/`.
- If the species under consideration has a multiple locus sequence type available, `phylogeny/{snp_caller}/cgMLST/bwa/phylogeny_with_st.svg` can be generated with each sample tagged with its Sequence Type.
- Phylogenies can also be generated over the full sequence of the reference used for mapping, for instance by calling `phylogeny/{snp_caller}/full_genome_{reference_id}/bwa/phylogeny_with_st.svg`

- If you do not want or cannot use core genome schemes, `typing/{snp_caller}/full_genome_33148/bwa/distances_in_snp_mst_no_st.svg` will show the minimum spanning tree over the full genome of the assembly ID `33148` (*S. aureus COL* genome from NCBI).
- If you want to genotype with mapping over one of your own sequenced sample, `typing/{snp_caller}/full_genome_S10_assembled_genome/bwa/distances_in_snp_mst_no_st.svg` will show the minimum spanning tree when mapping onto the sample called `S10`.

CHAPTER 8

All Deliverables

- assembly
- virulence
- resistance
- epidemiology