

Pathogen Genomic Epidemiology 2025

Faculty: Angela McLaughlin, Emma Griffiths, Finlay Maguire, Gary Van Domselaar, Idowu Olaw

November 24-26, 2025

Contents

I	Introduction	5
1	Workshop Info	7
1.1	Pre-work	7
1.2	Class Photo	7
1.3	Schedule	7
2	Meet Your Faculty	9
3	Data and Compute Setup	13
4	Module 1 Data Curation and Data Sharing	15
4.1	Lecture	15
4.2	Lab	15
5	Module 2 Emerging Pathogen Detection and Identification	17
5.1	Lecture	17
5.2	Lab	17
6	Module 3 Pathogen Typing	19
6.1	Lecture	19
6.2	Lab	19
7	Module 4 Outbreak Analysis	35
7.1	Lecture	35
7.2	Lab	35

8	Module 5 Phylodynamics	49
8.1	Lecture	49
8.2	Lab	49

Part I

Introduction

Chapter 1

Workshop Info

Welcome to the 2025 Pathogen Genomic Epidemiology Canadian Bioinformatics Workshop webpage!

1.1 Pre-work

1.2 Class Photo

Coming soon!

1.3 Schedule

Chapter 2

Meet Your Faculty

2.0.0.1 Angela McLaughlin

Postdoctoral Fellow Dalhousie University and University of Guelph
Burnaby, BC, Canada
— ez928230@dal.ca

Angela McLaughlin is a postdoctoral fellow in Dr. Finlay Maguire’s lab at Dalhousie University, in collaboration with Dr. Zvonimir Poljak at University of Guelph. Her research interests span viral phylogenetics (HIV-1, SARS-CoV-2, and influenza virus), genomic epidemiology, bioinformatics, public health, wildlife surveillance, and statistical/machine learning/mathematical models of pathogen transmission. Her current project aims to predict host specificity of avian influenza virus H5Nx using machine learning models of viral genomic features with phylogenetics-informed cross-validation and hierarchical segment to whole genome ensemble models.

2.0.0.2 Emma Griffiths

Research Associate, Faculty of Health Sciences Simon Fraser University Vancouver, BC, Canada
— emma_griffiths@sfu.ca

Emma Griffiths is a research associate at the Centre for Infectious Disease Genomics and One Health (CIDGOH) in the Faculty of Health Sciences at Simon Fraser University in Vancouver, Canada. Her work focuses on developing and implementing ontologies and data standards for public health and food safety genomics to help improve data harmonization and integration. She is a member of the Standards Council of Canada and leads the Public Health Alliance for Genomic Epidemiology (PHA4GE) Data Structures Working Group.

2.0.0.3 Finlay Maguire

Assistant Professor, Faculty of Computer Science and Department of
Community Health & Epidemiology, Dalhousie University Halifax,
NS, Canada

— finlay.maguire@dal.ca, finlaymagui.re

Finlay Maguire is a genomic epidemiologist whose work centers on leveraging data in innovative ways to answer questions related to applied health and social issues. This includes developing bioinformatics methods to more effectively use genomic data to mitigate infectious diseases and broad interdisciplinary collaborations in areas such as refugee healthcare provision and online radicalisation. They are an active contributor to the national and international public health responses to emerging viral zoonoses and antimicrobial resistance, co-chair of the PHA4GE data structures working group, and act as a Pathogenomics Bioinformatics Lead for Sunnybrook's Shared Hospital Laboratory.

2.0.0.4 Gary Van Domselaar

Chief, Bioinformatics, National Microbiology Laboratory Public
Health Agency of Canada Winnipeg, MB, Canada

— gary.vandomselaar@phac-aspc.gc.ca

Dr. Gary Van Domselaar, PhD (University of Alberta, 2003) is the Chief of the Bioinformatics Section at the National Microbiology Laboratory in Winnipeg Canada and Associate Professor in the Department of Medical Microbiology and Infectious Diseases at the University of Manitoba. Dr. Van Domselaar's lab develops bioinformatics methods and pipelines to understand, track, and control circulating infectious diseases in Canada and globally. His research and development activities span metagenomics, infectious disease genomic epidemiology, genome annotation, population structure analysis, and microbial genome wide association studies.

2.0.0.5 Idowu Olawoye

Postdoctoral Associate University of Western Ontario London, ON,
Canada

— iolawoye@uwo.ca

Idowu is a postdoctoral associate in the Guthrie Lab at the University of Western Ontario. His research focuses on utilizing computational biology to understand transmission patterns, genomic evolution, and antimicrobial resistance of bacterial pathogens. He has been involved in numerous bioinformatics workshops and trainings across Africa and most recently in Canada.

2.0.0.6 Jennifer Guthrie

Assistant Professor University of Western Ontario London, ON,
Canada

— jennifer.guthrie@uwo.ca

Dr. Jennifer Guthrie is a Canada Research Chair in Pathogen Genomics and Bioinformatics and Assistant Professor in the Departments of Microbiology & Immunology and Epidemiology & Biostatistics at Western University; she also serves as an Adjunct Scientist at Public Health Ontario. A genomic epidemiologist by training, in her research Dr. Guthrie uses interdisciplinary approaches combining principles from data science and bioinformatics, and more traditional epidemiology and microbiology methods to further our understanding of disease transmission dynamics, antimicrobial resistance, and epidemiological characteristics of pathogens. Her research involves pathogens of public health importance such as SARS-CoV-2, influenza, *Mycobacterium tuberculosis*, and methicillin resistant *Staphylococcus aureus*.

2.0.0.7 Zhibin Lu

Senior Manager, Digital Research University Health Network
Toronto, ON, Canada — zhibin@gmail.com

Zhibin Lu is a senior manager at University Health Network Digital. He is responsible for UHN HPC operations and scientific software. He manages two HPC clusters at UHN, including system administration, user management, and maintenance of bioinformatics tools for HPC4health. He is also skilled in Next-Gen sequence data analysis and has developed and maintained bioinformatics pipelines at the Bioinformatics and HPC Core. He is a member of the Digital Research Alliance of Canada Bioinformatics National Team and Scheduling National Team.

2.0.0.8 Charlie Barclay

MSc Graduate Student Researcher University of British Columbia
Vancouver, BC, Canada

— cbarcl01@mail.ubc.ca

Charlie is an ontology curator at the Centre for Infectious Disease Genomics and One Health (CIDGOH), focusing on data standards for contextual data of genomic epidemiology, including wastewater surveillance. With five years of experience in data management, specializing in biodiversity and genomics data, she also actively contributes to the Public Health Alliance for Genomic Epidemiology (PHA4GE) and the Global Alliance for Genomics and Health (GA4GH).

Chapter 3

Data and Compute Setup

3.0.0.1 Course data downloads

Coming soon!

3.0.0.2 Compute setup

Coming soon!

Chapter 4

Module 1 Data Curation and Data Sharing

4.1 Lecture

4.2 Lab

Chapter 5

Module 2 Emerging Pathogen Detection and Identification

5.1 Lecture

5.2 Lab

Chapter 6

Module 3 Pathogen Typing

6.1 Lecture

6.2 Lab

6.2.1 Introduction

The scope of this practical session is to perform a core-genome multi-locus sequence typing (cgMLST) analysis, a genome-based molecular typing method widely adopted for genomic surveillance of bacterial species.

Due to the widespread adoption of high-throughput whole-genome sequencing (WGS) and the advancement of this technology, it has been incorporated in many surveillance programs such as PulseNet Canada to monitor foodborne outbreaks.

This practical is split into two sessions:

1. cgMLST allele schema creation all through to allele calling, which will be conducted on your terminal to generate necessary input files for the next session
2. Clustering of cgMLST data from chewBBACA and visualization in RStudio

In this exercise, you will begin by generating cgMLST allele calls for a collection of 200 *Staphylococcus aureus* genomes (50 complete and 150 draft genomes) from NCBI RefSeq and BVBRC.

6.2.2 PART ONE: chewBBACA Objective

- To create wgMLST and cgMLST schema for a collection of 200 MRSA *Staphylococcus aureus* genomes (50 complete genomes and 150 draft genomes) from NCBI RefSeq and BVBRC

6.2.2.1 Activating Conda environment

Before we begin, the software we need to conduct our analysis have already been installed on your instance using Conda. However, we need to activate this environment to use the necessary tools.

```
# Activate conda environment to use chewBBACA and Prodigal
conda activate chewie
```

6.2.2.2 Create training file using Prodigal

ChewBBACA requires a training file to predict genes and proteins for our species of interest. For this purpose, we are going to use the *S. aureus* USA300 reference genome to generate the training file.

```
prodigal -i USA300ref.fasta -t Staphylococcus_aureus.trn -p single
```

6.2.2.3 Create wgMLST schema using the complete genomes from NCBI

Now that we have our training file, we will start by creating a whole-genome multi-locus sequence typing (wgMLST) schema based on the complete 50 *S. aureus* genomes from NCBI RefSeq

```
chewBBACA.py CreateSchema -i genomes/NCBI-RefSeq-50 -o mrsa_schema --ptf Staphylococcus
```

This will use the prodigal training file to identify CDS from the complete set of *S. aureus* genomes in NCBI and save them in the mrsa_schema folder

The schema seed will be found in that folder with 3,389 identified loci.

6.2.2.4 Allele Calling

Next is to perform allele calling with the wgMLST schema that we created in the previous step.

This will determine the allelic profiles of the strains we are analyzing by identifying novel alleles, which will be added to the schema.

```
chewBBACA.py AlleleCall -i genomes/NCBI-RefSeq-50 -g mrsa_schema/schema_seed -o wgMLST_50 --cpu 4
```

Using a BLAST score ratio (BSR) of 0.6 and clustering similarity of 0.2, 13,762 novel alleles were identified, bringing the number of alleles in the schema to 17,151.

6.2.2.5 Paralog Detection

If you noticed from the previous results, 13 paralogs were identified. You can also find them in the `paralogous_counts.tsv` in the `wgMLST_50` folder. It is important to remove these paralogs from the schema due to their uncertainty in allele calls. To do this run the following

```
chewBBACA.py RemoveGenes -i wgMLST_50/results_alleles.tsv -g wgMLST_50/paralogous_counts.tsv -o w
```

By doing this, the new allelic profile without the paralogs are now saved in `results_alleles_NoParalogs.tsv`, which encompasses 3,376 loci

6.2.2.6 cgMLST schema analysis

After some QC, we can now determine how many loci are present in the core genome based on the allele calling results. This is based on a given threshold of loci presence in the genomes we analyzed. The `ExtractCgMLST` module uses 95%, 99%, and 100% to determine the set of loci in the core genome.

```
chewBBACA.py ExtractCgMLST -i wgMLST_50/results_alleles_NoParalogs.tsv -o wgMLST_50/cgMLST
```

If you look in the `cgMLST` folder, there is an interactive HTML line plot showing the cgMLST per threshold relative to the number of loci detected.

From the plot, how many loci are present in the core genome at 95%? We would use this threshold to account for loci that might not be identified due to sequencing coverage and assembly issues.

6.2.2.7 cgMLST assignment for 150 genomes

These are 150 MRSA genomes with good quality from Australia pulled from BVBRC with associated metadata. Mislabelled species have been excluded.

Using the 1,999 loci found in 95% core genomes, we will perform Allele assignment on the 150 genomes

```
chewBBACA.py AlleleCall -i genomes/BVBRC-150 -g mrsa_schema/schema_seed --gl wgMLST_50
```

At the end of the analysis, this added 13,244 novel alleles to the schema with no paralogs

6.2.2.8 Convert Allele Calls

To convert the allelic profiles into a suitable format that can be parsed into other tools, we need to convert the non-integers such as INF, ASM, PLOT3, and PLOT5 into integers

```
chewBBACA.py ExtractCgMLST -i BVBRC-150_results/results_alleles.tsv -o BVBRC-150_resul
```

6.2.3 PART TWO: cgMLST clustering Objective

The purpose of this lab session is to take the cgMLST allele assignment that we have previously generated with chewBBACA and cluster them using Hamming distance matrix and allele thresholds. In addition, we would visualize our dataset using a dendrogram and annotate it with our cluster threshold and associated metadata.

The overall goal is to infer genetic relatedness in our dataset by leveraging computational tools to investigate potential outbreaks and their potential sources.

6.2.3.1 Getting Started

We will begin by installing the necessary packages and helper script that we need to analyze our data.

Every time you begin a new R session, you must reload all the packages and scripts!

```
# install packages requiring BiocManager

if (!requireNamespace('BiocManager', quietly = TRUE))
```

```

install.packages('BiocManager')

if (!requireNamespace('ComplexHeatmap', quietly = TRUE))
  BiocManager::install('ComplexHeatmap')

if (!requireNamespace('treedataverse', quietly = TRUE))
  BiocManager::install("YuLab-SMU/treedataverse")

# install other packages

required_packages <- c("tidyverse", "data.table", "BiocManager", "plotly", "ggnewscale", "circlize",
  "randomcoloR", "phangorn", "knitr", "purrr", "scales", "remotes", "reactable",
  "ggtreeExtra")

not_installed <- required_packages[!required_packages %in% installed.packages()[,"Package"]]

if (length(not_installed) > 0) {
  install.packages(not_installed, quiet = TRUE)
}

# load packages

suppressPackageStartupMessages(library(tidyverse))
suppressPackageStartupMessages(library(data.table))
suppressPackageStartupMessages(library(treedataverse))
suppressPackageStartupMessages(library(plotly))
suppressPackageStartupMessages(library(ggnewscale))
suppressPackageStartupMessages(library(ComplexHeatmap))
suppressPackageStartupMessages(library(circlize))
suppressPackageStartupMessages(library(randomcoloR))
suppressPackageStartupMessages(library(RColorBrewer))
suppressPackageStartupMessages(library(phangorn))
suppressPackageStartupMessages(library(knitr))
suppressPackageStartupMessages(library(purrr))
suppressPackageStartupMessages(library(scales))
suppressPackageStartupMessages(library(reactable))
suppressPackageStartupMessages(library(ggnewscale))
suppressPackageStartupMessages(library(ggtreeExtra))

## source helper R scripts
source("src/ggtree_helper.R")
source("src/cluster_count_helper.R")
source("src/cluster_helper.R")
source("src/cgmlst_helper.R")

```

Next read the cgMLST data from chewBBACA and the metadata into memory

```
# read cgMLST and metadata
aus_mrsa_cgMLST <- read.delim("BVBRC-150_results/cgMLST-150/cgMLST95.tsv", sep = "\t")

mrsa.data <- read.csv("MRSA_AUS_metadata.csv", header = F)
```

Having loaded our files, we need to make some adjustments to our metadata to make it suitable for our analysis.

first, we are going to change the header, then we are going to create a new column for the host to exclude the taxon ID

```
# change the header of the metadata file
new.header <- mrsa.data[2,]
aus_mrsa_metadata <- mrsa.data[-c(1,2),]
colnames(aus_mrsa_metadata) <- new.header

head(aus_mrsa_metadata)

aus_mrsa_metadata_reordered <- aus_mrsa_metadata %>%
  select(6, everything()) # move 6th column to the first position

head(aus_mrsa_metadata_reordered)

# Create a new Host column with just the host name
aus_mrsa_metadata_reordered <- aus_mrsa_metadata_reordered %>%
  mutate(Host = str_remove(`host (common name)`, "\\s*\\[.*\\]"))
```

Can you spot the differences between the two metadata? Which one do you think we are going to use?

6.2.3.2 Calculating the Hamming Distance

The traditional approach for cgMLST analysis is based on computing pairwise distances between allele profiles as a proxy for the underlying genetic similarity between two isolates. Below, you are introduced to a distance metric called the **Hamming distance**, which is based on computing the number of differences between a pair of character vectors. The Hamming distance is useful for comparing profiles where a majority of characters are defined, such as profiles comprising core loci.

Given two character vectors of equal lengths, the Hamming distance is the total number of positions in which the two vectors are different:

Profile A: [0 , 2 , 0 , 5 , 5 , 0 , 0 , 0 , 0]

Profile B: [0 , 1 , 0 , 4 , 3 , 0 , 0 , 0 , 0]

A != B: [0 , 1 , 0 , 1 , 1 , 0 , 0 , 0 , 0]

Hamming distance = `sum(A != B)` = 3

In phylogenetic analysis, distance-based approaches are rather flexible in the sense that they can be constructed from any measure that estimates genetic similarity through the direct comparison of data points. In cgMLST, we compare allele profiles. In Mash, we compare k-mer profiles.

In the context of two cgMLST profiles, the Hamming distance is calculated based on the number of allele differences across all loci as a proportion of total number of loci evaluated.

Hamming distances will be computed in an *all vs. all* fashion to generate a pairwise distance matrix that will subsequently serve as the input for distance-based tree-building algorithms. Run the following code chunk:

```
# use hamming helper function to compute pairwise Hamming distance of cgMLST data

# compute Hamming distance
# PATIENCE!!! this step might take several minutes depending on the size of the dataset

mrsa.dist_mat <- aus_mrsa_cgMLST %>%
  column_to_rownames("FILE") %>%
  t() %>%
  hamming()

# the dimension should be symmetric and should be the size of dataset (i.e. number of QC-passed g

dim(mrsa.dist_mat)
```

6.2.3.3 Hierarchical Clustering of cgMLST profiles by Hamming distance

In this section we will be performing hierarchical clustering of the Hamming distance matrix using the Unweighted Pair Group Method with Arithmetic Mean (i.e. **UPGMA**) algorithm.

Let's cluster the Hamming distance matrix by running the code chunk below:

```
# compute hierarchical clustering with hclust function and complete linkage method

mrsa.hc <- mrsa.dist_mat%>%
  as.dist() %>%
  hclust(method = "complete")

# reorder distance matrix according to the hc order

mrsa.dist_mat <- mrsa.dist_mat[mrsa.hc$order, mrsa.hc$order]

# write reordered distance matrix to files
dir.create("output/clusters", recursive = T)
write.table(mrsa.dist_mat, file = "output/clusters/dist_mat_ordered_cgmlst.tsv",
  quote = F, row.names = F, sep = "\t")
```

6.2.3.4 Cluster extraction at multiple distance thresholds

Identifying clusters of genomes sharing highly similar cgMLST profiles through the application of distance thresholds is a common practice in genomic surveillance and epidemiological investigations. These genomic clusters can become “analytical units” that can be tracked across space and time:

- The detection of a novel genomic cluster comprising isolates from multiple human clinical cases can signal the emergence of an outbreak, thus requiring a public health response in order to contain further cases.
- An examination of the evolving genomic cluster over time can provide important epidemiological insights on outbreak progression.
- The co-clustering of outbreak isolates with isolates from food/environmental sources can assist epidemiologists investigating an outbreak by linking the outbreak isolates to isolates from possible sources/reservoirs of the pathogen.

Identifying cluster membership for every isolate in the dataset at multiple distance thresholds gives us the analytical flexibility to define suitable thresholds for analyzing the pathogen in question. From a practical perspective, a threshold that is too fine-grained will yield a large proportion of the dataset in singleton clusters, making it difficult to link outbreak isolates to one another and to potential outbreak sources. Conversely, using a threshold that is not fine-grained enough will fail to fully exploit the discriminatory power of genomic data, grouping outbreak and non-outbreak isolates indiscriminately.

Adjusting similarity thresholds for cluster membership can be used to tweak the granularity of clusters: higher distance threshold produce larger clusters whereas lower distance thresholds will produce

smaller clusters. A threshold of 0 will generate clusters with **identical** profiles.

For membership at all possible distance thresholds, run the following code chunk:

```
# Define genomic cluster membership at all possible distance thresholds

mrna.cgmlst_loci = (ncol(aus_mrsa_cgMLST)-1) ## maximum distance
interval = 1 ## edit interval of interest; currently set to 1 then change to 10 to see how many

threshld <- seq(0, mrna.cgmlst_loci,interval)

# extract cluster membership across multiple thresholds
cluster_group <- map(threshld, function(x) {
  mrna.hc %>%
    cutree(h = x) %>%
    as.factor()
})

# create name for each threshold
names(cluster_group) <- paste0("Threshold_", threshld)

# print clustering results table, no duplicated names are allowed
(
  all.clusters <- data.frame(cluster_group ) %>%
    rownames_to_column("FILE")
)

# reactable(clusters_all)

# write file of genomic cluster memberships at multiple thresholds
write.table(all.clusters, file = "output/clusters/clusters_all.tsv",
  quote = F, row.names = F, sep = "\t")
```

Questions: How many distinct thresholds are theoretically possible for this dataset? How would you determine the maximum number of allele differences across all genomes in the dataset based on the table above? *hint: at some threshold, all genomes collapse to the same cluster...*

For membership at select distance thresholds, run the following code chunk:

```

# Define genomic cluster membership at user-defined distance thresholds

# Can either specify specific thresholds as a numeric vector i.e. c(0, 5, 10, 15)
# If you're feeling lazy, you can also specify using c(seq(5, 100, 5)), which stands for
# increments of 5". You can also string multiple notations together within the same number

threshld <-c(0, seq(5, 100, 5), seq(200, mrsa.cgmlst_loci, 100)) # currently reads as
                                                                # then from 5 to 100 in
                                                                # then from 200 to max.
                                                                # in increments of 100

# extract cluster membership across multiple thresholds
cluster_group <- map(threshld, function(x) {
  mrsa.hc %>%
  cutree(h = x) %>%
  as.factor()
})

# create name for each threshold
names(cluster_group) <- paste0("Threshold_", threshld)

# print clustering results table, no duplicated names are allowed
(
  new_defined_clusters <- data.frame(cluster_group) %>%
  rownames_to_column("FILE")
)

# write file
write.table(new_defined_clusters, file = "output/clusters/new_defined_clusters.tsv",
  quote = F, row.names = F, sep = "\t")

```

Questions: How many user-defined thresholds have we generated in the settings provided here? Which cluster does genome “1280_21738” belong to at a threshold of 25 allele differences? How would you determine how many different clusters were generated at each particular threshold?

6.2.3.5 Distance matrix visualization via ordered heatmap

A very useful visualization when dealing with distance matrices involves performing clustering, arranging the entries of the matrix based on the clustering order, and displaying the similarity as a heatmap. This produces a heatmap with a distinct 45 degree axis in which clusters of profiles with significant similarity representing possible clades/lineages can be plainly seen as “pockets of

heat". In this section of the lab, we will visualize the distance matrix using the `ComplexHeatmap` package and we will be comparing these clades to the MLST genotype information on the isolates by overlaying MLST information on the heatmap, which will allow us to examine whether the organism's MLST is concordant with putative lineages defined by cgMLST similarity as displayed on the clustered heatmap.

Let's run the code chunk below:

```
# create column annotations for heatmap
# to display clade information
set.seed(123)

htmap_annot <- aus_mrsa_metadata_reordered$genotype
names(htmap_annot) <- aus_mrsa_metadata$`BV-BRC genome ID`

htmap_annot <- htmap_annot[order(factor(names(htmap_annot),
                                       levels = rownames(mrsa.dist_mat))))]

# create heatmap
mrsa.dist_mat %>%
  Heatmap(
    name = "cgMLST\nDistance",
    show_row_names = F, # do not display row labels
    show_column_names = F, # do not display column labels
    # use custom color gradient
    col = colorRamp2(
      c(min(mrsa.dist_mat), mean(mrsa.dist_mat), max(mrsa.dist_mat)),
      c("#f76c6a", "#eebd32", "#7ece97")
    ),
    # add column annotation to show MLST info overlaid on the clustered heatmap
    top_annotation = HeatmapAnnotation(
      MLST= htmap_annot,
      col = list(
        MLST = structure(brewer.pal(length(unique(htmap_annot)), "Set3"),
                        names = unique(htmap_annot))
      )
    )
  )
```

6.2.3.6 Annotated Dendrograms and cluster evaluation

Here you will convert hierarchical clustering result (`mrsa.hc`) into a dendrogram using the `as.phylo()` function from the `ape` package. To visualize the resulting

dendrogram, we will use the R package **ggtree**, which offers an extensive suite of functions to manipulate, visualize, and annotate tree-like data structures. In this section, you will be introduced to some of the different visual capabilities of **ggtree** and we will progressively update the same tree with several layers of visual annotations based on available metadata.

Note that there is a massive amount of information in the internet dedicated to **ggtree** and all of its capabilities for advanced visualizations. Here we will barely scratch the surface.

6.2.3.7 Circular vs. Rectangular dendrograms for simple tree visualization

Radial vs. Rectangular dendrograms are different ways of visualizing a tree. Radial trees are capable of displaying a lot of simple data in a smaller footprint. In contrast, rectangular trees can display more complex data in a visualization that is easy on the eyes. Rectangular trees rendered as pdf format allow you to really zoom into sub-branches of the tree in greater detail when viewed in a pdf reader or in a browser window. This is essential when you're dealing with larger datasets comprising hundreds of isolates such as the one we are dealing with here.

Run the following code chunks to:

1. Plot a circular tree of the entire dataset with the tree tips colored by MLST information. *You can assign a different metadata field to the `color_var` variable to update the mapping of the color aesthetics in the tree. For example setting `color_var = "Host"` will color the tree tips by source of isolate.*
2. Plot the same tree as a rectangular tree.

```
# convert hierarchical clustering to a dendrogram
set.seed(123)
mrna.cg_tree <- as.phylo(mrna.hc)
```

```
# also, write out the dendrogram to a Newick tree file, can be imported into ITOL or o
dir.create("output/trees", recursive = T)
write.tree(mrna.cg_tree, file = "output/trees/tree_complete_linkage.newick")
```

```
# visualization with a color variable using R ggtree
color_var <- "genotype" # editable variable, currently set to "MLST".
```

```
## filter out unknown serotypes to define the number of colors needed for remaining se
n_colors <- length(unique(pull(aus_mrna_metadata_reordered, !!sym(color_var))))
```

```
## create circular dendrogram with variable-colored tippoints
cg_tree_cir <- mrsa.cg_tree %>%
  ggtree(layout='circular', # set tree shape
    size = 1 # branch width
  )%<+% aus_mrsa_metadata_reordered +
  geom_tippoint(aes(color = as.factor(!sym(color_var))),
    size = 2,na.rm=TRUE) +
  guides(color = guide_legend(title = "MLST", override.aes = list(size = 3) ) ) +
  scale_color_manual(values = distinctColorPalette(n_colors),na.value = "grey")

cg_tree_cir

# Create rectangular tree with MLST tip points and text tip labels

# create tip labels
aus_mrsa_metadata_reordered <- aus_mrsa_metadata_reordered %>%
  mutate(tip_lab = paste0(Host,"/ ",sample_collection_date))

## plot
set.seed(123)
mrsa.cg_tree %>%
  ggtree(layout= "rectangular")%<+% aus_mrsa_metadata_reordered +
  geom_tippoint(aes(color = as.factor(!sym(color_var))),
    size = 3,na.rm=TRUE) +
  geom_tiplab(aes(label = tip_lab),
    offset = 5,
    align = TRUE,
    linetype = NULL,
    size = 3) +theme_tree2()+
  geom_treescale(y = 130, x = 0.2) +
  guides(color = guide_legend(title = "MLST", override.aes = list(size = 3) ) ) +
  scale_color_manual(values = distinctColorPalette(n_colors))

# We save to pdf format
ggsave(file = "output/trees/clade_subtree_rec.pdf", height = 45, width = 40)
```

Questions: Between ST22 and ST93, which one shows more heterogeneity based on the cgMLST data?

6.2.3.8 Superimposing genomic cluster information onto dendrograms

Let's now superimpose the genomic cluster information on the previous dendrograms (Radial & Rectangular) to examine whether the above code chunk for extracting cluster memberships at various thresholds has generated sensible cluster assignments. Run the code chunk below to insert text labels that span across tree tips assigned to the same clusters at a specified threshold. Interchange your threshold to 50 and 15 and see how it affects your cluster outcome.

You can edit the `target_threshold` variable to examine how cluster membership changes in response to clustering distance cutoffs.

Run the code chunk below for the radial tree:

```
# Radial tree visualization

# assign all clusters to clusters for visualization
mrса.clusters <- all.clusters

target_threshold <- 25 # Editable variable, currently set to a threshold of 25.

aus_mrsa_metadata_reordered <- aus_mrsa_metadata_reordered %>%
  select(-tip_lab)

# variable to subset clusters
target_variable <- paste0("Threshold_", target_threshold)

# create cluster group list object
cluster_grp <- mrса.clusters %>%
  select(FILE, target_variable) %>%
  group_by(!sym(target_variable)) %>%
  {setNames(group_split(.), group_keys()[[1]])} %>%
  map(~pull(., FILE))
# sequester singleton clusters
cluster_grp <- cluster_grp[which(map_dbl(cluster_grp, ~length(.)) > 10)]

# create clade group list object
meta2 <- aus_mrsa_metadata_reordered %>%
  filter(genotype != "unknown")
Clade_grp <- meta2 %>%
  select(`BV-BRC genome ID`, genotype) %>%
  split(f = as.factor(.$genotype)) %>%
  map(~pull(., `BV-BRC genome ID`))
```



```

# add cluster memberships and clade information to tree object
mrsa.cg_tree <- groupOTU(mrsa.cg_tree, cluster_grp, 'Clusters')
mrsa.cg_tree <- groupOTU(mrsa.cg_tree, Clade_grp, 'MLST')

# plot core genome tree where colored blocks = clusters and text annotations = clades
valid_clade_grp <- Clade_grp %>%
  keep(~all(.x %in% mrsa.cg_tree$tip.label)) %>%
  keep(~length(.x) > 1) # Need at least 2 tips for MRCA

valid_cluster_grp <- cluster_grp %>%
  keep(~all(.x %in% mrsa.cg_tree$tip.label)) %>%
  keep(~length(.x) > 1)

mlst.t25 <- mrsa.cg_tree %>%
  ggtree(layout='circular', # Editable tree shape, currently set to circular?
    size = 1 # branch width
  ) +

# add colored blocks to display clades
geom_highlight(
  mapping = aes(
    node = node,
    fill = MLST,
    subset = node %in% map_dbl(
      valid_clade_grp,
      ~getMRCA(mrsa.cg_tree, .)
    )
  )
) +

#add text annotations to display clusters
geom_cladelab(
  mapping = aes(
    node = node,
    label = Clusters,
    subset = node %in% map_dbl(
      valid_cluster_grp,
      ~ getMRCA(mrsa.cg_tree, .)
    )
  ),
  horizontal=T,
  angle = 'auto',
  barsize = 0.75,
  offset = 50,
  offset.text = 50,
  align = T
) +

```

```

# legend parameters
guides(fill = guide_legend(
  nrow = 11,
  override.aes = list(alpha = 0.8)
) +
labs(fill = "MLST") +
scale_fill_brewer(palette = "Paired")

mlst.t25

final_plot <- mlst.t25 %<+% aus_mrsa_metadata_reordered +
  geom_tippoint(aes(color = Host), size = 2) +
  scale_colour_manual(name = "Host", values = c("#008080", "#ffa500", "#00ff00", "#000000"))

final_plot

```

Here we can see a couple of things:

1. The cgMLST cluster threshold of 25 allele differences has more discriminatory power than MLST for investigating putative outbreaks or inferring genetic similarity
2. Cluster 50 comprises of mostly horse isolates and two human isolates
3. We can infer the directionality of the outbreak in cluster 50 by looking at the sample collection dates and location if the data are present

```
ggsave("Threshold25_clusters.pdf", plot = final_plot, height = 10, width = 8, device = "pdf")
```

Congratulations! You have successfully completed Module 3.

Chapter 7

Module 4 Outbreak Analysis

Author: Finlay Maguire

Last Modified: 2025-11-11

7.1 Lecture

[Link to PDF of slides from google](#)

7.2 Lab

1. Background
2. Setup
3. Cluster Identification
4. SNP Analysis
5. Transmission Inference

In this lab practical we will be using microbial genomes (and associated contextual metadata) to investigate a suspected nosocomial (hospital-acquired) outbreak of *Methicillin resistance Staphylococcus aureus* (MRSA) in a UK-based Neonatal Intensive Care Unit (NICU). The data we are using comes from a classic study which demonstrated the utility of whole genome sequencing for investigating an MRSA outbreak within a Special Care Baby Unit (also known as a NICU) of the Cambridge University Hospitals NHS Foundation Trust (CUH).

Harris SR, Cartwright EJP, Török ME, et al. Whole-genome sequencing for analysis of an outbreak of methicillin-resistant *Staphylococcus aureus*: a descriptive study. *Lancet Infect Dis* 2012 10.1016/S1473-3099(12)70268-2

This practical will step you through the genomic components of a typical nosocomial bacterial outbreak investigation, specifically:

- Identifying potential outbreak clusters
- Determining SNP distances
- Inferring outbreak phylogenies
- Performing transmission inference

7.2.1 Background

Staphylococcus aureus is a Gram-positive coccoidal bacteria that forms a normal commensal member of the microbiome in 20-40% of people (Karsten, 2017). However, when skin and mucosal barriers are disrupted (e.g., through medical procedures such as catheterisation, tracheal tubes, surgeries) *S. aureus* can cause opportunistic invasive infections (Lee, 2018).

Since the 1960s, variants have emerged and spread globally that are resistant to the majority of β -lactam antibiotics through the independent acquisitions of the staphylococcal cassette chromosome *mec* (SCC*mec*) (IWG-SSC, 2009). This combination of commensal carriage, antimicrobial resistance, and healthcare-associated infection opportunities (along with additional virulence factors) have led to MRSA becoming a leading cause of hospital-associated mortality and morbidity globally (GBD, 2024). Therefore, tracking and preventing MRSA outbreaks is a major priority for infection prevention and control (IPAC or IPC) within hospital settings especially within the NICU due to the vulnerability of immunocompromised preterm infants and the high frequency of invasive procedures. This has led to adoption of genomic epidemiological approaches to detect MRSA outbreaks and track transmission links not apparent from traditional approaches alone (Blane, 2024). These sort of outbreak investigations are generally led by the infection prevention & control (IPAC or IPC) team in a clinical setting or by the relevant public health team depending on the suspected infection source, geographic scale, and regional resource availability (e.g., Public Health Agency of Canada/Canadian Food Inspection Agency or provincial bodies like Public Health Ontario).

Within the SCBU/NICU of Cambridge University Hospitals NHS Trust (and most hospitals) all inpatients are screened for MRSA carriage upon admission and once per week thereafter using culture-based or nucleic-acid based test. Specific IPAC policies and guidelines will determine criteria for initiating an outbreak investigation (e.g., a certain number of positive screens within a specific time and/or location).

Routine surveillance has identified that 3 infants within the SCBU/NICU (P11-P13) are positive for MRSA carriage at the same time. Isolates from these 3 infants have also been shown to have the same pattern of antibiotic resistances. The IPAC team has therefore been activated to further investigate this as a suspected outbreak in the SCBU/NICU at CUH. They have performed a systematic review of all MRSA isolates from the SCBU/NICU over the preceding 6-12 months and identified a series of overlapping MRSA carriages with this same set of resistances. Due to multi-week gaps in the positive SCBU/NICU isolates they have also collected potentially matching MRSA isolates from parents and the wider hospital/community. To gain better insight into this potential outbreak and identify which of these isolates are linked by direct transmission chains, CUH have sequenced these isolates using 150bp paired-end reads via an Illumina MiSeq platform.

You are a clinical bioinformatician who has been asked to support the investigation and analyses these genomes

7.2.2 Set-Up

This lab practical will involve running the following software:

- mlst
- poppunk
- ska
- gubbins
- IQTree
- GraphSNP

Typically, most of these analyses would be run using a workflow such as bactopia that you are confident will reproducibly generate validated and verified outputs (ideally as part of your institution's ISO15189 accreditation or equivalent). We will run the tools directly today so you get insight into what these workflows are actually doing!

7.2.2.1 Data

Using ssh connect to the provided analysis server instance:

```
ssh -i YOUR_KEY.pem YOUR_USER@XX.uhn-hpc.ca
```

Now create a folder in your `~/workspace` and copy/link over the files you will need:

```
mkdir -p ~/workspace/module4
cd ~/workspace/module4
cp -r ~/CourseData/module4/contextual_data.csv ~/CourseData/module4/SASCBU26_reference
ln -s ~/CourseData/module4/assemblies .
```

Specific collection dates were not available so have been inferred based on relative sampling gaps

When you are finished with these steps you should be inside your work directory `~/workspace/module4`. You can verify this by running the command `pwd`. **s**
Output after running pwd

```
~/workspace/module4
```

You now see an `assemblies/` folder in the current directory along with a `contextual_metadata.tsv` (all the contextual data needed for this analysis) and `SASCBU26_reference.fna` (the reference genome we will use later):

Output after running ls

```
SASCBU26_reference.fna  assemblies  contextual_data.csv
```

7.2.2.2 Activate environment

Next we will activate the pre-installed conda environment, which will have all the tools needed by this tutorial pre-installed. To do this please run the following:

Commands

```
conda activate outbreak
```

You should see the command-prompt (where you type commands) switch to include `(outbreak)` at the beginning, showing you are inside this environment. You should also be able to run the `mlst` command like `mlst --version` and see output:

Output after running `mlst --version`

```
mlst 2.23.0
```

7.2.2.3 Find your IP address

Similar to yesterday, we will want to either use the assigned hostname (e.g., `xx.uhn-hpc.ca` where `xx` is your instance number) or find the IP address of your machine on AWS so we can access some files from your machine on the web browser. To find your IP address you can run:

Commands

```
curl http://checkip.amazonaws.com
```

This should print a number like `XX.XX.XX.XX`. Once you have your address, try going to `http://xx.uhn-hpc.ca` or `http://IP-ADDRESS` and clicking the link for **module4**. This page will be referred to later to easily download/view some of our output files.

7.2.3 Cluster Identification

All isolates that are putatively linked to the outbreak have been done so on the basis of time, location, and phenotype (specifically antibiotic susceptibilities). However, just because 2 isolates have the same resistance pattern does not mean they are closely related. Although this is a relatively small set of isolates we are often evaluating very large numbers of genomes for their potential connection to an outbreak. Additionally, most downstream outbreak analyses (such as transmission inference) perform best when applied to as little diversity as possible (i.e., just genomes from a single outbreak event).

This means our first task is to identify and group which isolates are actually connect

are likely to be linked using the sort of typing methods you encountered in the previous lab practical. This helps us to eliminate unrelated genomes and determine whether 1 or more outbreaks are likely to be taking place.

To make things faster for this practical, we have provided pre-inferred genome assemblies (generated using shovill with the skesa assembler).

You can infer the 7-gene MLST using the pubMLST *Staphylococcus aureus* scheme by running the `mlst` tool on each of the genome assemblies.

```
mkdir -p outbreak_cluster_identification; cd outbreak_cluster_identification
mlst --scheme saureus ../assemblies/*.fa > mlst.tsv
```

Now download the `mlst.tsv` output file (using either `scp` or the IP address listed above) and open it in your tabular data tool of choice (e.g., pandas/python, R, excel).

Based on these results, which genomes do you suspect may not be part of this outbreak?

- P27 & P28 have different MLSTs (ST1 and ST8) to any other genome so unlikely to be linked to an outbreak.
- P29 & P34-38 (with P37 a partial match) could be an ST22 outbreak.
- P30-33 could be an ST772 outbreak.
- All remaining isolates belong to a large ST2731 set

Why might genomes with a different MLST still be closely related or those with the same MLST be relatively unrelated?

MLST are only 7 genes so a small amount of mutation (or conservation!) in just these genes can lead to a totally different MLST - schemes are designed to try and be robust but are mutation is a random process with high variance!

We could solve this challenge using a more fine-grained scheme like cg/wgMLST (as discussed in the previous module) but we are going to use **poppunk** in this lab.

poppunk is a rapid k-mer based genome clustering tool that groups genomes together based on both their core and accessory genome distances. This allows differential clustering of similar genomes which have acquired novel plasmids and so on.

The developers of **poppunk** provide some pre-computed databases (although it is relatively easy to create your own) for different species which are available here.

We have already downloaded a reference database for you, so all you have to do is prepare a 2-column file **genomes.txt** which has a series of names in the first column and the path to the related genome assembly in the other. We can do this using a quick bash one-line loop:

```
for i in ../assemblies/*.fa; do isolate=$(echo $i | cut -d '/' -f3 | cut -d '.' -f1);
```

Then we can run the following to get **poppunk** cluster assignments:

```
poppunk_assign --db ~/CourseData/module4/staphylococcus_aureus_v1_full --query genomes
```

Now download/inspect **poppunk/poppunk_clusters.csv** file

Which genomes does this analysis suggest should be excluded? Do they match the MLST results?

P27-28 are assigned to cluster 1 and P30-33 are assigned cluster 12. Remaining samples are assigned to cluster 3.

- P27 and P28 were different singleton STs for MLST that share a few alleles so supports eliminating them
- P30-33 were ST772 so poppunk and MLST agree and support eliminating these samples (although they could be their own distinct outbreak)
- ST22 and ST2731 isolates were both assigned to cluster 3. If we look carefully at the MLST profile for these STs in `mlst.tsv` we can see that these STs only differ by a single allele (`arcC`) so could be linked in the main outbreak.

We can't be sure about cluster 3 without further analysis but we can drop cluster 1 and 12 for now.

7.2.4 SNP Analysis

Now that we have eliminated some isolates that are unlikely to be connected to our main outbreak we can do a deeper analysis of the evolutionary relationships between our isolates. For this we are going to perform a phylogenetic analysis. There are several way this could be done:

- Inferring and aligning the core genome of our isolates by running a tool like `panaroo` on the genome assembly annotation files (generated with `prokka` or `bakta`).
- Mapping individual reads against a reference genome with a tool like `snippy`.
- Mapping SNPs between genomes using `ska` (then using a reference to order these SNPs)

For SNP analyses, why might we want use a reference derived from an outbreak isolate instead of standard species reference genome?

We maximise the number of detectable SNPs by using a reference as close as possible to our other genomes

For this lab, we are going to do the last option using a complete high quality reference genome generated from this outbreak subsequent to the original manuscript: SASCBU26

First let's generate the input file for SKA using the poppunk results

```
cd ~/workspace/module4; mkdir -p ska_phylogeny; cd ska_phylogeny
paste <(grep "3$" ../outbreak_cluster_identification/poppunk/poppunk_clusters.csv | cut -d, -f1)
echo -e "SASCBU26\t../SASCBU26_reference.fna" >> ska_input
```

Now we need to generate an SKA index:

```
ska build -f ska_input -k 31 -o ska_index --threads 4
```

Then map these split k-mers against the reference (required to order them for the next step):

```
ska map -o ska.aln --ambig-mask ../SASCBU26_reference.fna ska_index.skf
```

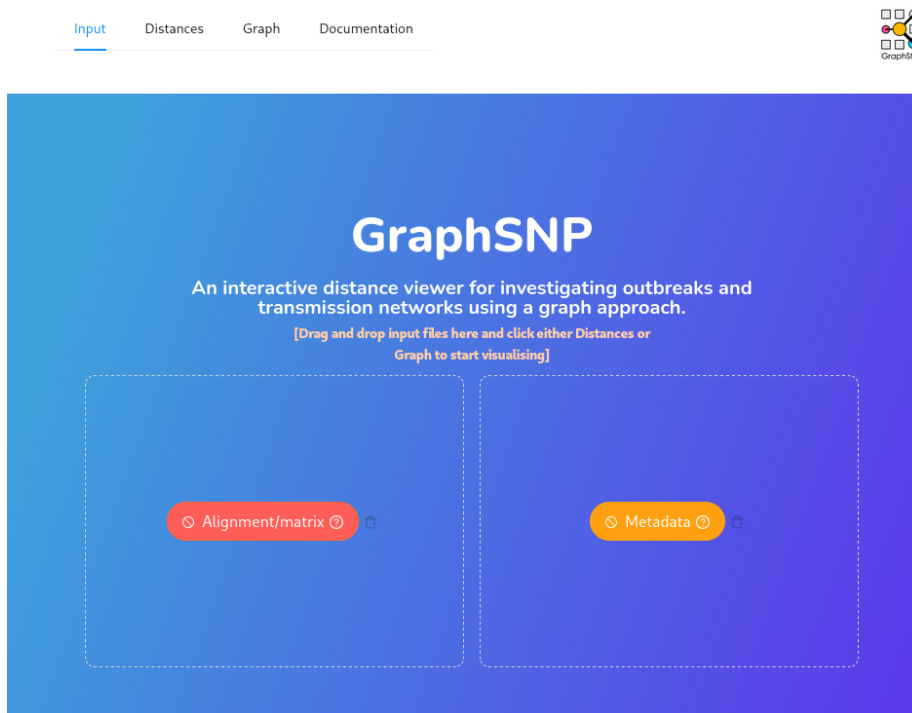
This gives us an alignment but due the distorting effects of recombination we typically want to try and remove any potential recombinant sites from the alignment. There are several tools that can help us do this (`verticall`, `ClonalFrameML`, `gubbins`) and today we are going to use `gubbins`.

```
mkdir -p gubbins  
run_gubbins.py --prefix gubbins/gubbins ska.aln  
mask_gubbins_aln.py --aln ska.aln --gff gubbins/gubbins.recombination_predictions.gff
```

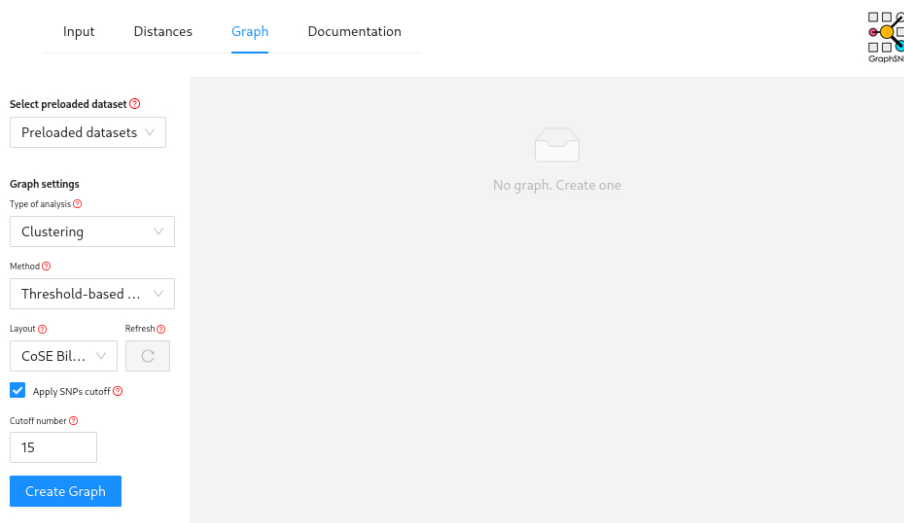
Now that we have a masked alignment `gubbins.masked.aln` we can analyse the SNP network using `GraphSNP` to further refine our potential outbreak samples. First, generate a SNP distance matrix using `snp-dists`:

```
snp-dists gubbins.masked.aln | sed 's/\\t/,/g' > raw_outbreak_matrix.csv
```

Now download the `raw_outbreak_matrix.csv`, navigate to `GraphSNP` in your browser, and upload the `raw_outbreak_matrix.csv` as your Alignment/matrix.



Then click on “Graph” at the top of the page (may be hidden behind 3 horizontal lines on smaller screens), increase your “Cutoff number” to 50, and then hit the “Create Graph” button.



This will create a minimum spanning tree grouped using a 50 SNP cut-off.

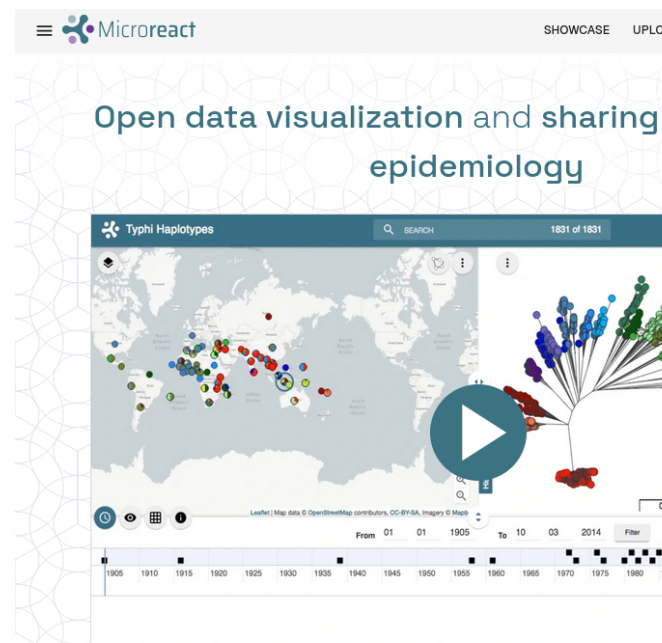
Which genomes do you think you can eliminate from being part of these immediate outbreak cluster?

Based on the graph, isolates P29 & P34-P38 can be eliminated. These were the ST22 isolates identified by `mlst` that `poppunk` convinced us to look at a bit more closely. We can see the closest is only ~80 SNPs from our outbreak cluster.

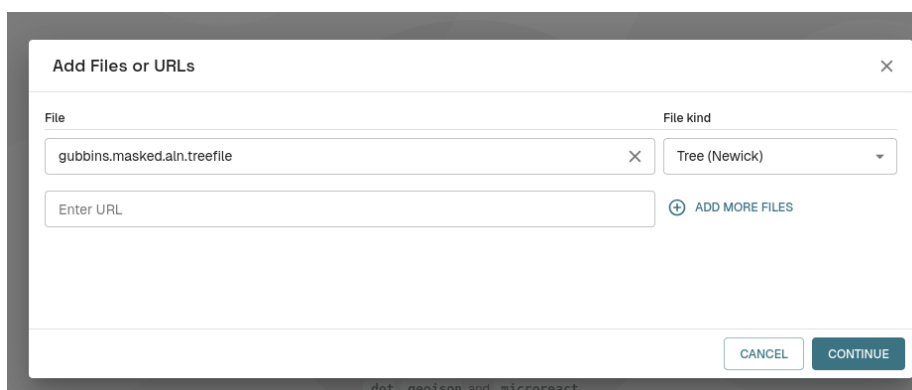
Let's also look at these isolates using a traditional phylogenetic approach. Go back to the server and run the following command to build a maximum-likelihood tree using `iqtree`:

```
iqtree -s gubbins.masked.aln
```

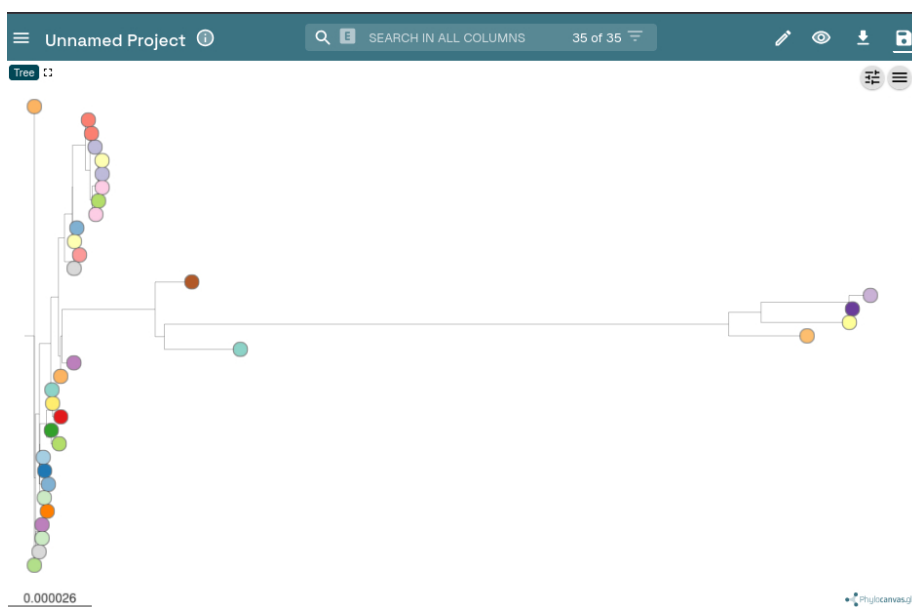
This will generate a lot of outputs but if you download the newick formatted phylogeny `gubbins.masked.aln.treefile` and go to `microreact` we can visualise the tree.



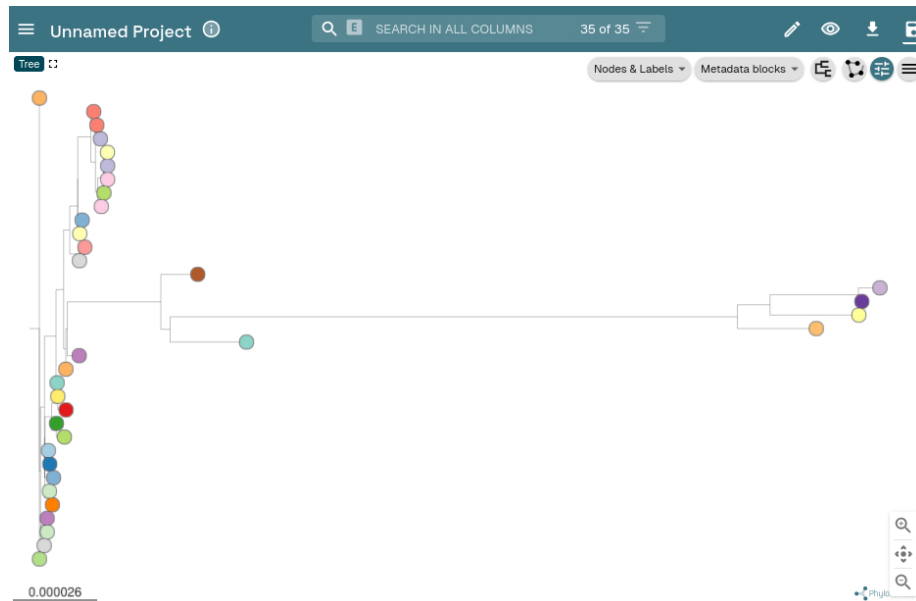
Hit the “Upload” button and select your treefile.



Then hit the settings slider icon at the top right



Finally, select radial/unrooted tree.



Does the phylogeny show the same genomes as being outside the main outbreak cluster

Yes, P37 & P38 are close to the main outbreak cluster and could be included but for now we will exclude them.

Now we have refined our final set of suspected outbreak isolates we can create our final alignment of just these sequences.

Create a SNP distance matrix and phylogeny for just the final outbreak isolates by repeating the above steps with a modified `ska_input` (if you create a new folder you'll avoid getting mixed up with your new datas).

A pre-computed distance matrix and phylogeny can be copied from `~/CourseData/module4/refined_outbreak_matrix.csv` and `~/CourseData/module4/refined.gubbins`

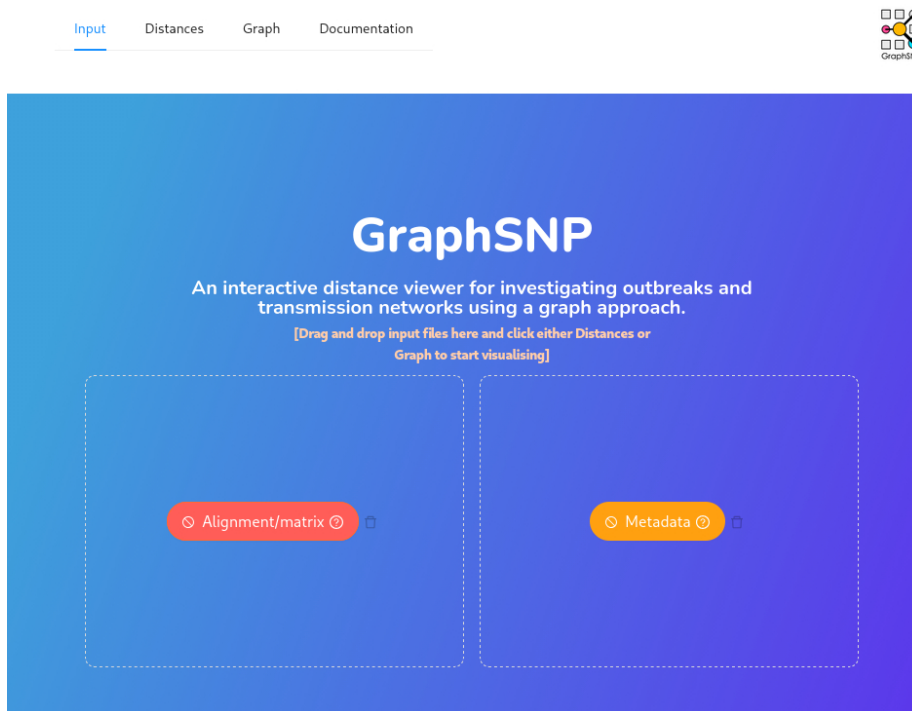
Transmission Inference

With our refined outbreak distance matrix and phylogeny we are going to perform some transmission analyses by combining these with some of our context data.

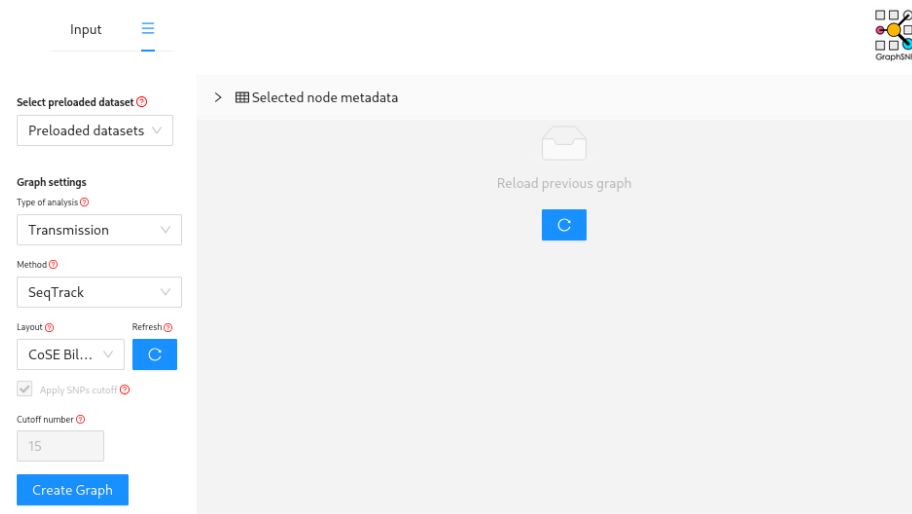
First we are going to use the SeqTrack method as implemented in GraphSNP.

To do this download `contextual_metadata.csv` and your refined distance matrix (e.g., `refined_outbreak_matrix.csv` or whatever you have named it) and then navigate to GraphSNP.

This time you must upload the new distance matrix under Alignment/matrix and the metadata table under Metadata.



Then navigate to “Graph” but this time select analysis type “transmission” and click “Create Graph”



Which person is inferred to have infected the largest number of other patients?

P8 or P4

Which baby/babies was/were inferred to be infected by a parent?

P13 and P15

Congratulations you've performed your first transmission analysis!

The next step would be to try and perform a more sophisticated transmission inference method using **TransPhylo**. However, this would require us to infer a time-scaled phylogeny and you'll discover more about how to do that in the next module!

Chapter 8

Module 5 Phylodynamics

8.1 Lecture

8.2 Lab