LegioCluster Tutorial

10/01/2020

This three step tutorial will walk you through all the steps needed to set up and run the LegioCluster pipeline. The pipeline is already set up to run *Legionella pneumophila*, but for this tutorial, we add *Streptococcus pyogenes* as a new species and analyze two sets of reads in different operating modes.

Prerequisites
Required software:
- Linux/Unix/Mac operating system
- Docker
- Python 3.7 or higher
- the Python packages `pydot`, `numpy` and `matplotlib`, which can be installed with:

```
pip install <package-name>
```

All reference and read files needed to run the tutorial are located in the `/Tutorial/` folder. The pipeline was designed to analyze paired-end Illumina reads files.

Installation
Download and uncompress the `LegioCluster.tar.gz` file. In the `/LegioCluster/` folder, you will find three links to PY files:

`add_species.py` allows users to modify the pipeline to analyze additional bacterial species. Requires at least one reference genome (FASTA file) for that species.

`LegioCluster.py` starts with an interactive menu to properly format all required input data into an input TXT file. The best way to start LegioCluster.

`LegioCluster_main.py` lacks the interactive menu and requires the name of an input TXT file as parameter, which should have been saved to the `/input/` folder.

Tutorial overview
In Part 1, we will add *S. pyogenes* to the list of species the pipeline can use, which involves uploading (or copying) one or more reference genome files and answering a few simple questions.

In Part 2, we run the pipeline in automated mode, which demonstrates how the pipeline works under normal circumstances. The pipeline will process reads from two isolates and compare them to a reference to call SNPs and indels events (= mutation events). For the first isolate, Spy_sample_1, that reference will be the genome sequence we provided in Tutorial Part 1, *S. pyogenes* M1_GAS. Since the Spy_sample_1 and strain M1_GAS are rather different based on the number of mutation events, the pipeline will automatically generate a new cluster centered on Spy_sample_1. The second isolate, Spy_sample_2, will be mapped to the new reference Spy_sample_1 and added to that cluster.

In Part 3, we address custom options that can be helpful under specific circumstances.

- Over-riding the reference selection result, thereby forcing an isolate to be mapped to a specific reference. This can be useful if a number of distinct isolates need to be compared to each other.
- Forcing an isolate to become a new cluster / candidate reference. This can be useful if an isolate was placed automatically into a cluster, but additional data indicate that this isolate should form a separate cluster.
- Renaming isolates. The pipeline automatically selects the isolate's name based on the name of the read file. This can be overwritten by the user and is required when trying to run an isolate through the pipeline for a second time, since the pipeline keeps track of isolate names to prevent over-writing existing files.
- Adding metadata for individual isolates. Each isolate can be given its own set of up to three metadata. Over-writing the reference or designating an isolate as new reference will take up one metadata slot.

A note about the Docker images

By default, LegioCluster pulls Docker images when it starts and deletes all images and containers after it is done. That's because some images, like Kraken, take up a lot of resources. This feature can be turned off (at your own risk) by changing line 63, "RESET_DOCKER = True" to "RESET_DOCKER = False", in /Py_code/LegioCluster_main.py.

Time requirements

Part 1 should take only a few minutes. Depending on your computer, Parts 2 and 3 should take about 20 - 30 minutes each, most of which is time required by the pipeline to analyze the read files and generate the results. Annotated examples of Reports are included in the `/Tutorial/` folder to show the user what kind of output to expect.

If you have and questions or comments, please feel free to contact us:

Wolfgang Haas, Ph.D.

Wadsworth Center
New York State Department of Health
120 New Scotland Ave.
Albany, New York 12208
Email: wolfgang.haas@health.ny.gov

Tutorial Part 1:
Adding species to the pipeline

The LegioCluster pipeline can be used with any bacterial species after adding one or more reference genome(s) for that species. The `add_species.py` module was designed to automate this step by generating or updating all corresponding files and folders. All the user needs to do is to upload the FASTA file(s) with the reference genome(s) and answer a few questions. Shown below is the process for *Streptococcus pyogenes*; other species might require (slightly) more user input.

1. Upload one or more reference genomes

At least one reference genome is needed to check for contamination and to map the query's reads prior to SNP/indel calling. In addition, each reference will become a cluster of related isolates, so it is important that any two references are not too similar to each other. You can use any FASTA file, including those you generated yourself or down-loaded from NCBI (e.g.: https://www.ncbi.nlm.nih.gov/genome/?term=streptococcus+pyogenes%5Borgn%5D). If you are using files downloaded from NCBI, uncompress the file and give it the name of the strain and the extension ".fna", then upload or copy the reference FASTA file(s) into the `/LegioCluster/` folder.

For this tutorial, go to the `/LegioCluster/` folder, and unzip, rename, and copy the files for the two reference genomes from the `/Tutorial/` to the `/LegioCluster/` folder:

```
./LegioCluster$ gunzip Tutorial/GCF_000006785.2_ASM678v2_genomic.fna.gz
./LegioCluster$ mv Tutorial/GCF_000006785.2_ASM678v2_genomic.fna ./M1_GAS.fna
./LegioCluster$ gunzip Tutorial/GCF_000743015.1_ASM74301v1_genomic.fna.gz
./LegioCluster$ mv Tutorial/GCF_000743015.1_ASM74301v1_genomic.fna ./ATCC_19615.fna
```

2. Start the `add_species.py` module

In the `/LegioCluster/` folder, enter:

```
./LegioCluster$ python3 add_species.py
```

This starts the program with a welcome screen:

```
Welcome to the LegioCluster Species Addition program!
```

3. Enter a species name (and species abbreviation, if needed)

Next you will see a list of those species that can already be processed by the pipeline. If your species is already on the list, enter "`Q`" to exit. Otherwise, enter the complete scientific name.

```
The following species can be processed by the pipeline:
-------------------------------------------------------
Lpn:  Legionella pneumophila

Please enter a genus and species name (e.g. "Escherichia coli")
 'Q' to exit
Genus and species name:
Streptococcus pyogenes
```

The pipeline assigns each species a unique three-letter species abbreviation, such as "Spy" for ***S****treptococcus* ***py****ogenes*. In the event that an abbreviation is already in use, the user will be asked to select a different abbreviation.

4. Upload one or more reference genomes for that species

Assuming that you uploaded or copied at least one reference file into the /LegioCluster/ folder, press ENTER, then provide the file names when prompted. Please note that the first strain on the list will represent the entire species when searching for contamination, so this strain should be typical for the entire species.

```
Press ENTER after you have uploaded the reference genome file(s).
ENTER

Please enter the names of the reference genome files below.
 'Q' to exit
Reference file name(s):
M1_GAS.fna ATCC_19615.fna
```

5. If necessary, change the median genome length

The median genome length is used for quality control purposes and to calculate read coverage. For the most common pathogens, this number has already been extracted from the NCBI database and will be applied automatically. Otherwise, the genome length is calculated based on the reference genome(s) that you just uploaded. In that case, you have the option to replace the calculated number with one that is statistically more robust (e.g. taken from NCBI):

```
Based on the genomes submitted, the median genome length is:  1852433

You can replace this number, e.g. with one from NCBI's Taxonomy browser.
 'Y' to accept the median genome length
 'N' to replace it
 'Q' to exit
Accept median genome length?:
N
Please enter the median genome length: 1953456
```

6. Confirm your input

Check that your input is correct and select "**Y**" to continue or enter "**N**" to exit and start over.

```
Please confirm the following data are correct:
-------------------------------------------------------
Species name:                Streptococcus pyogenes
Species abbreviation:        Spy
List of reference files:     ['M1_GAS.fna', 'ATCC_19615.fna']
Median genome length:        1831320

     'Y' to complete the species addition
     'N' to exit without adding the species
    Continue?:
    Y
```

7. Next steps.

After the user enters "Y", the program will generate a file that keeps track of all isolates that have been processed (to prevent duplications), generate folders for the species and each reference strain, reformat the headers in the FASTA files, save the old `config.py` file under a new name and produce a new, updated `config.py` file.

```
     Made subfolders and a "prev_isolates.txt" file for Spy/

     Made subfolders for:
     M1_GAS.fna
     ATCC_19615.fna

     Reformatted headers in M1_GAS.fna
     Reformatted headers in ATCC_19615.fna

     Saved the old config.py file as: config_200827_140127.py

     Updated the config.py file.

     ... and we are done!
```

Please note that the old `config.py` file will be saved under a new name, using the date and time stamp as a suffix. Once completed, you should be able to process read files from the new species by the LegioCluster pipeline. In Tutorial Step 2, we will analyze two sets of Illumina paired-end read files from a *S. pyogenes* sequencing project.

Tutorial Part 2:
Normal operations

Now that *Streptococcus pyogenes* has been added to the list of species the pipeline can process, we use two sets of read files to demonstrate how the pipeline operates. The pipeline will process the reads and compare them to the best matching reference to call SNPs and indels events (= mutation events, ME). For the first isolate, Spy_sample_1, that reference will be selected from the two references that we uploaded in Part 1 of the Tutorial.

Since neither one of the two references is a close match, Spy_sample_1 will be used to start a new cluster and will be added to the list of candidate reference genomes. The second isolate, Spy_sample_2, will then have three references to choose from, including the newly added Spy_sample_1. Since the two isolates are very similar, Spy_sample_2 will be mapped to Spy_sample_1 and added to the Spy_sample_1 cluster. Minimum spanning trees and ME matrices will show how the isolates in the cluster differ from each other. A phylogenetic tree will be generated for each cluster with three or more isolates.

In this example, we show you how to submit two samples together, but you can just as well submit 20 or 200 samples at once (although the `logging.txt` file tends to get very large in the latter case). In Part 3 of the Tutorial we show you how to submit samples individually to make sample-specific adjustments.

1. Upload or copy the read files

Copy the read files for `Spy_sample_1` and `Spy_sample_2` into the `/LegioCluster/reads/Spy/` folder. All read files are kept in the `/LegioCluster/reads/` folder, sorted by species, here `/Spy/`. For the Tutorial, copy the files from the `/Tutorial/` into the `/reads/Spy/` folder.

```
./LegioCluster$ mv Tutorial/Spy_sample* reads/Spy/
```

2. Start LegioCluster

The LegioCluster pipeline requires for each isolate the following information:
- the user's initials, which will become part of the output folder's name
- the species abbreviation (e.g. "Spy" for ***S****. **py**ogenes*)
- the name of the file with the forward reads (the name of the isolate and the name of the reverse read file will be deduced from the forward read file's name)

This information will be summarized in a TXT file that is fed into the actual LegioCluster pipeline.

Start the LegioCluster pipeline by entering

```
./LegioCluster$ python3 LegioCluster.py
```

This will print a welcome message and ask for the user's initials:

```
 Welcome to LegioCluster
 ------------------------------

 Please enter your initials
 'Q' to exit:
WH
```

The fastest way to submit a large number of samples is to enter all relevant information only once. Enter "Y" to continue with the mass submission; we will cover individual submissions, which provides more options, in the third tutorial.

```
'Y' to submit multiple isolates together
'N' to submit isolates individually
'Q' to exit
Submit multiple isolates together?:
Y
```

Next, you will be asked for the species abbreviation. Note that "Spy" has been added as one of the options. Getting this step wrong will mean that none of the read files will pass the contamination check done by the pipeline.

```
Please select a species abbreviation from the list below.
 'Lpn', 'Spy'
 'Q' to exit
Species abbreviation:
Spy
```

The read file suffixes might vary with the core facility that is doing the sequencing. Here, the default is '`_R1_001.fastq.gz`' and '`_R2_001.fastq.gz`'. The information is used to extract the name of the reverse read file and the isolate name.

```
The pipeline accepts only paired Illumina reads.
The default suffix for all FORWARD reads is '_R1_001.fastq.gz'.
The default suffix for all REVERSE reads is '_R2_001.fastq.gz'.
 'A' to accept the defaults
 'C' to change the defaults
 'Q' to exit
Read file suffixes:
A
```

By default, the pipeline will automatically select the best mapping reference from all available reference candidates. However, sometimes it might be necessary to select one specific reference for all isolates, in which case you should select the "**S**" option. Beware, this will force isolates into the same cluster that might not belong together, so it's a good idea to back up

your /Genomes/, /References/, and /VCF_files/ folders. Select "**A**" to let the pipeline choose a reference automatically; we will cover the "**S**" option in Tutorial Part 3.

```
Reference genome selection
 'A' (recommended): let the pipeline select a reference automatically
 'S' manually select a specific mapping reference
 'Q' to exit
Selection:
A
```

You can enter meta data, such as outbreak number, source, dates, etc, in up to three fields. Note that here, these fields will be applied to all isolates submitted.

```
Enter metadata (optional, 1/3): tutorial part 2

Enter metadata (optional, 2/3): multiple submissions

Enter metadata (optional, 3/3): normal operations
```

When entering the names of the read files, you only need to supply the names of the forward reads, separated by white space (no commas). All samples need to be of the same species. The program will look for these files in the species-specific reads folder, in this case: /LegioCluster/reads/Spy/.

```
Enter the names of all FORWARD read files, separated by single
spaces:
Spy_sample_1_R1_001.fastq.gz Spy_sample_2_R1_001.fastq.gz
```

Next, you will be given a chance to check what you have submitted and asked to provide the name of a file that will be used as input to the actual LegioCluster pipeline. If you don't like what you see, select "**Q**" and start over. Otherwise, enter the name of a TXT file. This file will be stored in the /input/ folder for future use.

```
Please check your submission:
['Spy', 'Spy_sample_1', 'Spy_sample_1_R1_001.fastq.gz',
'Spy_sample_1_R2_001.fastq.gz', 'tutorial_part_2',
'multiple_submissions', 'normal_operations']
['Spy', 'Spy_sample_2', 'Spy_sample_2_R1_001.fastq.gz',
'Spy_sample_2_R2_001.fastq.gz', 'tutorial_part_2',
'multiple_submissions', 'normal_operations']


Please enter a file name ending in '.txt'
 'Q' to exit
File name:
tutorial_part2.txt
```

Final step: you can start the pipeline now by entering "**Y**" or exit by typing "**N**".

```
You can run the pipeline now or later, using 'python3
LegioCluster_main.py tutorial_part2.txt'
 'Y' to process all samples now
 'N' to exit
Do you want to run the pipeline now?:
Y
```

If you exited the program after saving the input TXT file, you can run the pipeline later by typing:

```
LegioCluster$ python3 LegioCluster_main.py tutorial_part2.txt
```

3. While LegioCluster is running

After you start the program, the following will happen:

a) Most of the required Docker images will be downloaded.

b) The reads for each isolate will be processed by the pipeline, one at a time. Data for samples that successfully completed will be copied to the `/output/` folder, those that failed will remain in the `/temp/` folder so the user can determine what went wrong. Most often, samples fail because of an insufficient number of reads or due to contamination.

c) After all samples have been processed, those that made it to the `/output/` folder will be used by Parsnp to generate a phylogenetic tree and to generate a summary CSV file. Parsnp will not run if there are less than three genomes in a cluster. In addition, all contigs generated by SPAdes for an isolate will be checked by Kraken to verify the species.

d) Finally, all Docker images and containers will be removed to free up memory.

4. The Results

If all went well, then the `/output/` folder should contain one CSV file, a TXT file and two folders. The folder names will consist of the user's initials, date, and time that the sample was analyzed, so no two folders should have the same name.

The CSV file

This file, such as `WH_Results_overview_1.csv`, will provide a quick overview of the results for each sample that successfully completed the pipeline. The prefix is the user's initials, while the suffix prevents files from being over-written. A detailed explanation of each column is given in **Table 1**.

Of special interest is the column **`Mash_References_and_Runner_up`**: Mash is used twice by the pipeline, once to check the reads for contamination and a second time to find the closest reference genome for mapping prior to SNP/indel calling, which is shown in this column.

For Spy_sample_1, the data indicate that strains M1_GAS and ATCC_19615 returned a similar number of matching hashes, so both were used for mapping. The next column, `Mapped_reads_[%]`, indicates that more reads were mapped to M1_GAS, making it the reference of choice (see `Mapping_reference`). Since the pipeline was run the first time for *S. pyogenes* and only two references were available, ATCC_19615 is listed a second time in `Mash_References_and_Runner_up`, this time as the runner-up. If ATCC_19615 had had the higher percentage of mapped reads, it would have been the reference of choice.

For Spy_sample_2, we see that Spy_sample_1 is not only on the list of references, but also the reference of choice. That's because Spy_sample_1 was so different from the other references available that it itself was added to the list of candidate references. Hence, Spy_sample_1 formed a new cluster with Spy_sample_2 as a member.

A second column of interest is labeled **`SNPs+indel_events_[N]`**, which uses four numbers, ME (IE, BID, SNP), to display the differences between the isolate and the reference strain.

- ME (mutation events) is the sum of indel events and SNPs
- IE is the number of indel events, where multiple (<= 100) insertions or deletions next to each other are counted as one event, regardless of the number of bases involved
- BID is the number of all bases involved in indels
- SNP is the number of Single Nucleotide Polymorphisms

The TXT file

The file `logging_1.txt` will contain a log of all commands used by the pipeline for each isolate. This information is helpful for determining which Docker image was used, the parameters applied, and any output returned.

Results folders

The folders for those samples that did not complete the pipeline will remain in the `/temp/` folder. The `report.txt` and `log.txt` files will usually give some information why the isolates failed to complete the pipeline; in most cases, it's due to low number of reads or contamination. All files, including temporary files, will be saved to help with the trouble-shooting. However, some of these files take up a lot of space, so it is good housekeeping to empty the `/temp/` folder occasionally.

The pipeline's results for all isolates that successfully completed will be copied from the `/temp/` to the `/output/` folder. **Table 2** shows an overview of all files produced by the pipeline per isolate.

The **`report.html`** file is the best place to get started, since it includes (almost) all results and imbedded figures. Annotated reports (in PDF format) are available in the `/Tutorial/` folder as examples. The **`report.txt`** file is a plain text version of `report.html`, but includes additional data after analyzing all contigs with Kraken to detect contigs of questionable origin (contamination, promiscuous plasmids, phages, etc).

The **`log.txt`** and **`logging.txt`** files includes a record of the Docker images that were used and the parameters that were applied. It might also include `stdout` and/or `stderr` information, but that can vary based on the Docker image. Should an isolate have failed to complete the pipeline, the `log.txt` file will most likely show the reason.

The number of mutation events and SNPs that separate individual isolates within a cluster are shown in the form of a matrix (**`ME_matrix.csv`**, `SNP_matrix.csv`) and Minimum Spanning Tree (**`MST_ME.png`**, `MST_SNP.png`). The pipeline provides both, ME and SNP data. The `muttaions_matrix.csv` file includes additional data, please see **Table 2** for details. Please note that in a MST, values are not additive: one should refer to one of the `matrix.csv` files to get information about two isolates that are not directly connected by an edge in the MST. The matrix and MST files will not be generated for new reference candidates, such as Spy_sample_1, since the cluster only contains one isolate.

Another file that shows how the new isolate relates to all other isolates in the same cluster is **`parsnp_tree.svg`**. Parsnp generates a phylogenetic tree based on the core genome of all isolates in a cluster. That core genome might look different from one cluster to another. In case a new genome was added to the list of candidate reference genomes, the phylogenetic tree will show how the candidate references are related to each other. Genomes that are significantly different from the others might be excluded from the tree. Parsnp requires at least three genomes to run, so there is no tree for Spy_sample_2, since the cluster contains only two isolates.

If an isolate was added to the list of candidate references, it is a good idea to check the **`kraken_res.txt`** and **`wrong_genus_contigs.fasta`** files to look for contigs of the wrong genus. The latter file can be uploaded directly to BLASTN at https://blast.ncbi.nlm.nih.gov/ to perform a nucleotide BALST search, if needed.

The *de novo* genome sequences assembled by SPAdes are stored in a file called **`SPAdes_contigs.fa`**, which contains all contigs. **`<ISO>.fa`**, where <ISO> is the name of the isolate, contains only those contigs that are >= 1 kb and have a coverage >= 7.5-fold. This file will be used to represent the isolate in phylogenetic trees or as a mapping reference. **`<ISO>_cc.fa`**, contains only those contigs that are >= 250 bp and have a coverage >= 3.0-fold; in addition, contigs whose Kraken results does not match the correct genus have been removed.

Please refer to **Table 2** for additional information.

Tutorial Part 3:
Individual submissions

Here we explore the options available when submitting each isolate individually, which allows for the selection of different species, changing an isolate's name, forcing the pipeline to use a specific mapping reference (instead of selecting the best matching one), or starting a new cluster based on the new isolate.

1. Upload or copy the read files

All read files should still be in the `/LegioCluster/reads/Spy/` folder, otherwise see part 2 of the Tutorial.

2. Start LegioCluster

Start the LegioCluster pipeline by entering

```
./LegioCluster$ python3 LegioCluster.py
```

As shown in Tutorial Part 3, this will print a welcome message and ask for the user's initials:

```
Welcome to LegioCluster
-----------------------------

Please enter your initials
'Q' to exit:
WH
```

Unlike Part 2 of the Tutorial, now we will submit data for each sample individually.

```
 'Y' to submit multiple isolates together
 'N' to submit isolates individually
 'Q' to exit
Submit multiple isolates together?:
N
```

Next, we will see the selection of available species again. Only this time, we will be asked to enter a species abbreviation for each set of read files, so we could combine "Lpn" and "Spy" into one submission, if we were so inclined.

```
Please select a species abbreviation from the list below.
 'Lpn', 'Spy'
 'Q' to exit
Species abbreviation:
Spy
```

Type '**A**' to accept the default settings for the suffixes.

```
The pipeline accepts only paired Illumina reads.
The default suffix for all FORWARD reads is '_R1_001.fastq.gz'.
The default suffix for all REVERSE reads is '_R2_001.fastq.gz'.
 'A' to accept the defaults
 'C' to change the defaults
 'Q' to exit
Read file suffixes:
A
```

The pipeline is designed to automatically select the best mapping reference (default). This time, enter "**S**" to select a specific reference. This can be useful to determine how different the current isolates are from another known outbreak cluster. We will cover the "**M**" option below.

```
Reference genome selection
 'A' (recommended): let the pipeline select a reference automatically
 'S' manually select a specific mapping reference
 'M' will make this isolate a candidate reference for subsequent isolates
 'Q' to exit
Selection:
S
```

Selecting "**S**" will show you the names of the available references: ATCC_19615 and M1_GAS were added when adding the species to the pipeline (Tutorial Step 1) and Spy_sample_1 was added by the pipeline (Tutorial Step 2). Here, we select M1_GAS.

```
Please select a reference genome from the list below.
 'ATCC_19615', 'M1_GAS', 'Spy_sample_1'
 'Q' to exit
Reference genome:
M1_GAS
```

Overwriting the reference takes up one of the three slots reserved for metadata, so we are left with only two remaining. Note that these fields will be applied to this one isolate only.

```
Enter metadata (optional, 2/3): Tutorial part 3: individual submissions

Enter metadata (optional, 3/3): override reference selection
```

Unlike the examples in Part 2, now you can submit only the name of a single forward read file.

```
Enter the name of the FORWARD read file:
Spy_sample_1_R1_001.fastq.gz
```

The pipeline automatically deduces the isolate's name from the name of the forward read file. If you prefer a different name, you can change it here. The pipeline also checks if an isolate has been processed before to avoid unnecessary duplications. To process a sample a second time, you need to change the isolate's name, as shown below.

```
You can change the isolate's default name: 'Spy_sample_1'.
 'Y' to enter a different name
 'N' to use 'Spy_sample_1'
 'Q' to exit
Do you want to change the isolate's name?:
Y
Please enter the new isolate name: Spy_sample_1_rerun
```

Next, you will be asked if you want to add more samples or if you are done with the sample submission:

```
Do you want to add more isolates?
 'Y' to add more isolates
 'N' to finish the sample submission
 'Q' to exit
Add more samples?:
Y
```

The next two steps are the same as for the first sample (although they wouldn't have to be).

```
Species abbreviation:
Spy

Read file suffixes:
A
```

Selecting "**M**" will add the current isolate to the list of reference genomes. This can be useful when an isolate was processed by the pipeline before, was designated as a new reference candidate, but then failed the quality control steps for new references. In these cases, the user can overwrite the QC steps that prevent an isolate from becoming a new reference.

```
Reference genome selection
 'A' (recommended): let the pipeline select a reference automatically
 'S' manually select a specific mapping reference
 'M' will make this isolate a candidate reference for subsequent isolates
 'Q' to exit
Selection:
M
```

As before, this takes up one of the metadata slots, so we are left with only two remaining. Note that these fields will be applied to this one isolate only.

```
Enter metadata (optional, 2/3): Tutorial part 3: individual submissions
Enter metadata (optional, 3/3): make new cluster
```

Submission of a single forward read file.

```
Enter the name of the FORWARD read file:
Spy_sample_2_R1_001.fastq.gz
```

As before, we want to change the isolate's name.

```
You can change the isolate's default name: 'Spy_sample_2'.
 'Y' to enter a different name
 'N' to use 'Spy_sample_2'
 'Q' to exit
Do you want to change the isolate's name?:
Y
Please enter the new isolate name: Spy_sample_2_rerun
```

That concludes the data submission for the second and last sample. To finish the submission process, type "**N**".

```
Do you want to add more isolates?
 'Y' to add more isolates
 'N' to finish the sample submission
 'Q' to exit
Add more samples?:
N
```

Check your submission. Note how for the first sample, the first metadata field specifies '`set_ref=M1_GAS.fa`', while it is '`make_ref`' for the second.

```
Please check your submission:
Please check your submission:
('Spy', 'Spy_sample_1_rerun', 'Spy_sample_1_R1_001.fastq.gz',
'Spy_sample_1_R2_001.fastq.gz', 'set_ref=M1_GAS',
'Tutorial_part_3:_individual_submissions', 'override_reference_selection')
('Spy', 'Spy_sample_2_rerun', 'Spy_sample_2_R1_001.fastq.gz',
'Spy_sample_2_R2_001.fastq.gz', 'make_ref',
'Tutorial_part_3:_individual_submissions', 'make_new_cluster')
```

Save the data to a TXT file, which will be saved in the `/input/` folder.

```
Please enter a file name ending in '.txt'
 'Q' to exit
File name:
tutorial_part3.txt
```

Final step: start the pipeline now by entering '**Y**'.

```
You can run the pipeline now or later, using 'python3
LegioCluster_main.py tutorial_part3.txt'
 'Y' to process all samples now
 'N' to exit
Do you want to run the pipeline now?:
Y
```

If you exited the program by typing "**N**", you can run the pipeline later as shown below. The program will search for the input TXT in the `/input/` folder

```
LegioCluster$ python3 LegioCluster_main.py tutorial_part3.txt
```

Comments:

The program will analyze each set of read files, as before. However, if you did not change the name of the isolates, the pipeline will terminate to prevent over-writing any existing files.

For the most part, the results will look similar to those produced during Tutorial part 2, except that `Spy_sample_1_rerun` will have been placed into the same cluster as strain M1_GAS instead of `Spy_sample_1`. `Spy_sample_2_rerun` will have formed its own cluster, instead of being placed into the same cluster as `Spy_sample_1`.

The examples show how the automatic clustering procedure implemented for the pipeline can be overwritten. This might be useful if all isolates need to be compared to one and the same reference or an isolate needs to be added as a new reference. In general, it is a good idea to use a second copy of the `/LegioCluster/` folder or backups of the `/References/`, `/Genomes/`, and `/VCF_files/` folders to separate automatically generated clusters from human-made clusters.

Table 1 Fields in a "Results_overview_1.csv" file

Column header	Description
Isolate_name	Name of the bacterial isolate
Folder_name	Name of the folder created by LegioCluster for that isolate
Pipeline_version	Version of the pipeline
Metadata_1	user supplied information "set_ref=<REF>" will designate <REF> as the reference for the isolate "make_ref" will designate the isolate as a new reference
Metadata_2	user supplied information
Metadata_3	user supplied information
F-read_file	Name of the forward Illumina read file
R-read_file	Name of the reverse Illumina read file
Trimmed_reads_[N] (1)	Number of paired Illumina reads after read trimming and removal of single or poor quality reads
Coverage	Fold read coverage, based on the number of reads, maximum read length, and the median genome length for that species
Percent_bases_Q30	Percent of bases with a quality score of Q30 or better
Mash_Species_and_Runner_up (1)	Species abbreviation and number of matching hashes after comparing the reads to reference genomes from various species. The top ranked species should be the same as the source of the reads. The score for the runner-up should be less than 20/400, otherwise the sample might be contaminated.
DNA_fragment_size_[med]	Median fragment size of Illumina products
Contigs_all_[N] (1)	Number of all contigs assembled de novo by SPAdes
Contigs_>1kb_[N]	Number of all contigs greater than 1 kb assembled by SPAdes
Poor_quality_contigs_[%]	Percentage of contigs that are either shorter than 1 kb and/or that have a coverage less than 7.5-fold.
Largest_contig_[bp]	Size in base pairs of the largest contig assembled by SPAdes
N50_[bp]	Size in base pairs of the contig that represents the N50 value
Genome_length_(all_contigs)[bp]	Length in base pairs of the genome when adding all contigs assembled by SPAdes
Genome_length_(contigs_>1kb)[bp]	Length in base pairs of the genome when adding only contigs > 1 kb assembled by SPAdes

Column header	Description
Number_of_gaps_[n]	Number of regions in the reference genome that are greater than 100 bp that are not covered by reads from the query (equivalent to large deletions in the query genome)
Sum_of_bases_in_gaps	Sum of all bases in the reference genome that are greater than 100 bp that are not covered by reads from the query
Mash_References_and_Runner_up	Reference strain names and numbers of matching hashes after comparing the contigs of the query to all candidate reference genomes for that species. The list will show one or more reference strain that are most similar to the query, as well as the runner-up.
Mapped_reads_[%] (2)	Percentage of query reads mapped to the best matching reference genome. If more than one number is shown, then the order of corresponding reference strains will be the same as in "Mash_References_and_Runner_up". If more than one reference is similar to the query, the one with the higher percentage of mapped reads will be selected.
Mapped_bases_in_ref_genome_[%]	Percentage of bases in the reference genome that have been mapped by reads. Corresponds roughly to the "Sum_of_bases_in_gaps".
Mapping_reference	Reference strain selected for read mapping prior to calling SNPs and indels. The query isolate will be added to the reference strain's cluster (or found a new cluster if the percentage of mapped reads is too low or the number of mutation events is too high).
SNPs+indel_events_[N] (2)	Four numbers in the form ME (IE, BID, SNP), where: - ME is the sum of indel events and SNPs (= mutation events) - IE is the number of insertion/deletion events, where one event can consist of one or multiple bases - BID is the number of all bases involved in indels - SNP is the number of Single Nucleotide Polymorphisms
Comments	Space for the user's comments

(1) The pipeline will terminate the analysis of the isolate if this value is not within threshold limits.

(2) If the percentage of mapped reads is less than 90% or the number of mutation events exceeds a species-specific threshold, then the query isolate will form a new cluster and will be added to the list of candidate reference strains for subsequent isolates.

Table 2 Overview of all files produced by the pipeline per isolate, where <ISO> represents the isolate's name.

File name	Description
report.html	Report in HTML format, showing the results produced by the pipeline, including various graphs and figures, but lacking data produced by Kraken species determination results for each contig.
report.txt	Report in TXT format, showing all results produced by the pipeline, including results from the Kraken species determination for each contig, but lacking graphs and figures.
log.txt	Lists all the logging data, including the commands used to run the Docker containers. Note that there is also a logging.txt file, with information from all samples run together.
mutations_matrix.csv (1)	Compares each isolate to every other isolate in the same cluster in the form ME (IE, BID, SNP), where ME = mutation events (IE + SNP) IE = insertions and deletion events BID = bases in insertions or deletions SNP = single nucleotide polymorphism see ME_matrix.csv for less complex data
ME_matrix.csv (1)	Compares each isolate to every other isolate in the same cluster based on the number of mutation events, where each ME is either an insertion/deletion event or a single nucleotide polymorphism. see mutations_matrix.csv for more detailed data.
SNP_matrix.csv (1)	Compares each isolate to every other isolate in the same cluster based on the number of single nucleotide polymorphisms (SNPs). see mutations_matrix.csv for more detailed data.
SPAdes_contigs.fa	All contigs assembled de novo by SPAdes
<ISO>.fa	Contigs assembled by SPAdes for isolate <ISO> that are >= 1 kb in length and have a coverage >= 7.5 fold
<ISO>_cc.fasta	Contigs assembled by SPAdes for isolate <ISO> that are >= 250 bp in length, have a coverage >= 3.0 fold, and that were of the correct genus as determined by Kraken
kraken_res.txt	Results from Kraken species determination: list of contig names and species determinations
wrong_genus_contigs.fasta	All contigs assembled by SPAdes that were of a genus (as determined by Kraken) that did not match the query's
distances_FAvNCBI.tab	Results from Mash for the query contigs versus all candidate reference genomes comparison; used to select one or more genomes for mapping

File name	Description
distances_RvSp.tab	Results from Mash for the query reads versus all species genomes comparison; used to determine gross contamination
freebayes_SNPs.vcf	List of all SNPs, insertions, deletions, and complex mutations detected by Freebayes that passed minimal filtering thresholds
Ampel_dist.png	Graph: contig length * coverage distribution. Green bars: high quality contigs >= 1 kb and with >= 7.5 fold coverage Yellow: contig length < 1 kb or coverage < 7.5 fold Red: contig length < 1 kb and coverage < 7.5 fold, suggesting poor data quality or contamination Black: contigs >= 1 kb and with >= 250 fold coverage, suggesting repeat regions or potential plasmids
MST_ME.png (1)	Graph: minimum spanning tree showing the fewest number of mutation events that connect one isolate to another isolate in the same cluster (use the ME_matrix.csv file to compare two isolates that are not directly linked in the MST)
MST_SNP.png (1)	Graph: minimum spanning tree showing the fewest number of SNPs that connect one isolate to another isolate in the same cluster (use the SNP_matrix.csv file to compare two isolates that are not directly linked in the MST)
contig_cov_dist.png	Graph: Coverage distribution per contig; blue line indicates the 7.5-fold cut-off.
contig_len_dist.png	Graph: Length distribution per contig; red line indicates the 1kb
histo_depths_1.png	Graph: Read depth per base; blue lines: mean +/- 3 * StdDev
mutation_dist.png	Graph: Distribution of SNPs and indels across the entire genome; each bar represents a 5 kb interval
per_base_quality_1.png	Graph: FastQC quality scores across all bases, forward reads
per_base_quality_2.png	Graph: FastQC quality scores across all bases, reverse reads
per_sequence_quality_1.png	Graph: FastQC quality score distribution over all sequences, forward reads
per_sequence_quality_2.png	Graph: FastQC quality score distribution over all sequences, reverse reads
plot_contig_cov.png	Graph: Contig coverage distribution: coverage of each contig; blue lines: 1 kb length and 7.5-fold coverage, respectively; red line: median coverage based on SPAdes output
plot_contig_len.png	Graph: Contig length distribution: length of each contig; horizontal lines: 100, 1000, etc kb blue vertical line: smallest contig > 1kb
plot_depths_1.png	Graph: Read depth per base: read depth across the entire genome; horizontal lines: mean +/- 3 * StdDev

File name	Description
`parsnp_tree.svg` (1)	Phylogenetic tree of all isolates in a cluster, generated by Parsnp; for new references, the new isolate relative to all other candidate references is shown

(1) This file might not be produced for all isolates, especially if there are not enough isolates in a cluster.