BioSeqDB Help Document

January 4, 2021
Arnie Berg

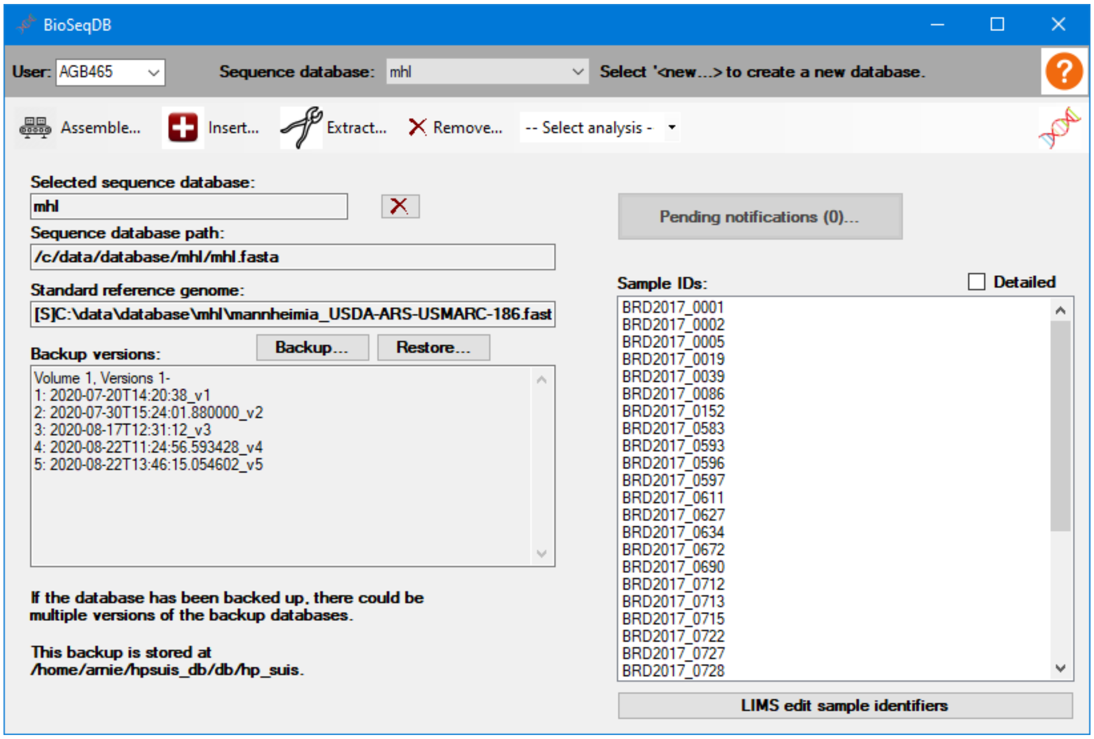
Table of Contents

Background

The purpose of the BioSeqDB program is to manage the process flow and information repository for sequence data of identified organisms. It runs as a Windows program and appears on the task bar as follows:

The BioSeqDB program is basically a wrapper that invokes Linux commands that have been developed to run in the Windows Subsystem for Linux (WSL) environment. The advantage of running BioSeqDB instead of the individual Linux commands is that BioSeqDB prompts the user for all the relevant parameters and functions needed to accomplish the biologically important objectives, and remembers previously selected options.

BioSeqDB maintains a separate database for each specified organism. If no existing database exists in which to insert a sequence for a different organism, a new database must be created. The BioSeqDB dialog always displays the currently selected sequence database, and optionally a standard reference genome associate with the database and a list of database backup versions. Databases may be selected from the 'Sequence Database' dropdown. The main BioSeqDB dialog also displays the Sample IDs from the currently selected database and appears as follows:



User: AGB465

Sequence database: mhl

Select '<new...> to create a new database.

Selected sequence database: mhl

Sequence database path: /c/data/database/mhl/mhl.fasta

Standard reference genome: [S]C:\data\database\mhl\mannheimia_USDA-ARS-USMARC-186.fast

Backup versions:

```
Volume 1, Versions 1-
1: 2020-07-20T14:20:38_v1
2: 2020-07-30T15:24:01.880000_v2
3: 2020-08-17T12:31:12_v3
4: 2020-08-22T11:24:56.593428_v4
5: 2020-08-22T13:46:15.054602_v5
```

If the database has been backed up, there could be multiple versions of the backup databases.

This backup is stored at /home/arnie/hpsuis_db/db/hp_suis.

Sample IDs: □ Detailed

```
BRD2017_0001
BRD2017_0002
BRD2017_0005
BRD2017_0019
BRD2017_0039
BRD2017_0086
BRD2017_0152
BRD2017_0583
BRD2017_0593
BRD2017_0596
BRD2017_0597
BRD2017_0611
BRD2017_0627
BRD2017_0634
BRD2017_0672
BRD2017_0690
BRD2017_0712
BRD2017_0713
BRD2017_0715
BRD2017_0722
BRD2017_0727
BRD2017_0728
```

BioSeqDB is designed as a multi-user system with clients accessing the BioSeqDB service (running on WIMMER) over the network. The databases reside on the server to be centrally available to all users.

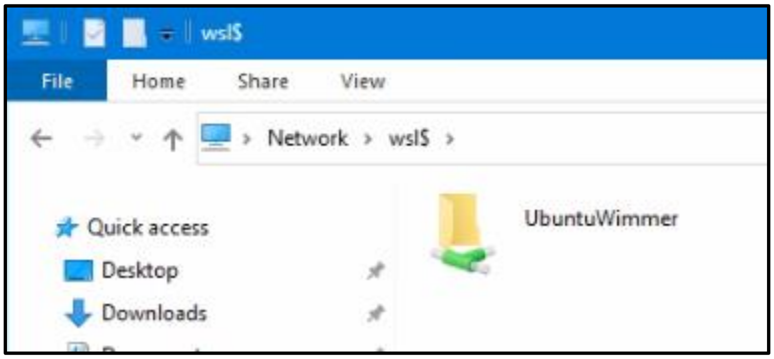
Most of this document describes the functionality available through this dialog. A section near the end provides some details for administrators of BioSeqDB. This document is meant to assist users in running BioSeqDB. It does not describe how to interpret results of the functions invoked by BioSeqDB.

File systems

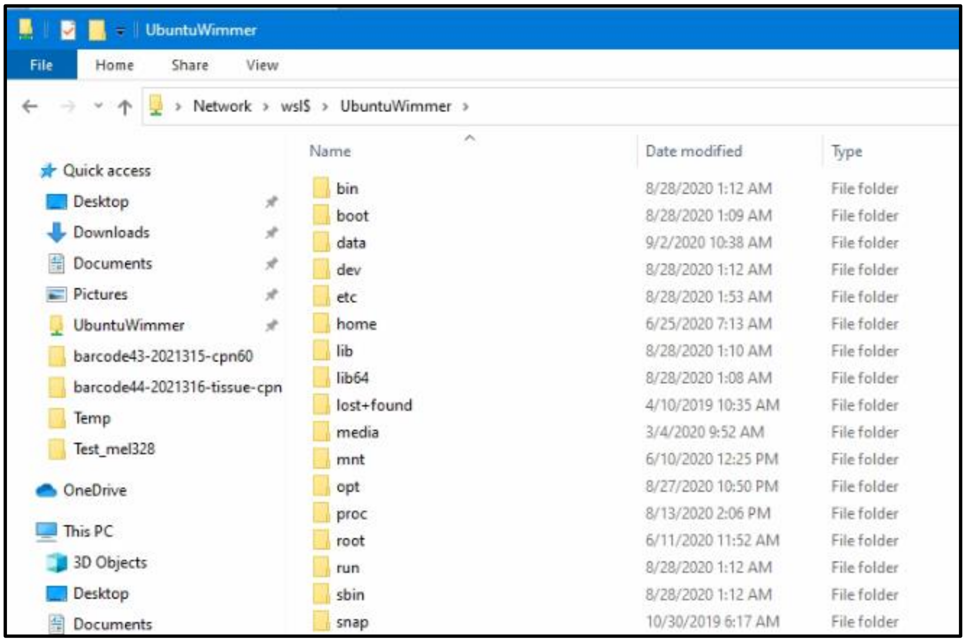
Through WSL, BioSeqDB bridges the gap between the Windows file system and the Linux file system.

If you are familiar with Linux, by using Windows Terminal in Windows you can start up the UbuntuWimmer shell (on WIMMER) as Linux user pds_wimmer to run in the Linux environment. Those details are hidden by BioSeqDB, but it still is important to be aware of the presence of the Linux file system and how it relates to the Windows file system.

From Windows File Explorer, the Linux file system may be accessed by entering '\\wsl$' in the address bar. This shows an instance of the Linux file system called UbuntuWimmer:



Double-click on UbuntuWimmer to open the top-level folder structure of the Linux file system:



Name	Date modified	Type
bin	8/28/2020 1:12 AM	File folder
boot	8/28/2020 1:09 AM	File folder
data	9/2/2020 10:38 AM	File folder
dev	8/28/2020 1:12 AM	File folder
etc	8/28/2020 1:53 AM	File folder
home	6/25/2020 7:13 AM	File folder
lib	8/28/2020 1:10 AM	File folder
lib64	8/28/2020 1:08 AM	File folder
lost+found	4/10/2019 10:35 AM	File folder
media	3/4/2020 9:52 AM	File folder
mnt	6/10/2020 12:25 PM	File folder
opt	8/27/2020 10:50 PM	File folder
proc	8/13/2020 2:06 PM	File folder
root	6/11/2020 11:52 AM	File folder
run	8/28/2020 1:12 AM	File folder
sbin	8/28/2020 1:12 AM	File folder
snap	10/30/2019 6:17 AM	File folder

If you are familiar with Windows, you should be aware of how Linux references folders in the Windows file system. This is important for BioSeqDB because although most functions run in Linux, most data are stored in the Windows file system. For example, all sequence data, both raw and assembled, are stored on the E: drive of WIMMER.

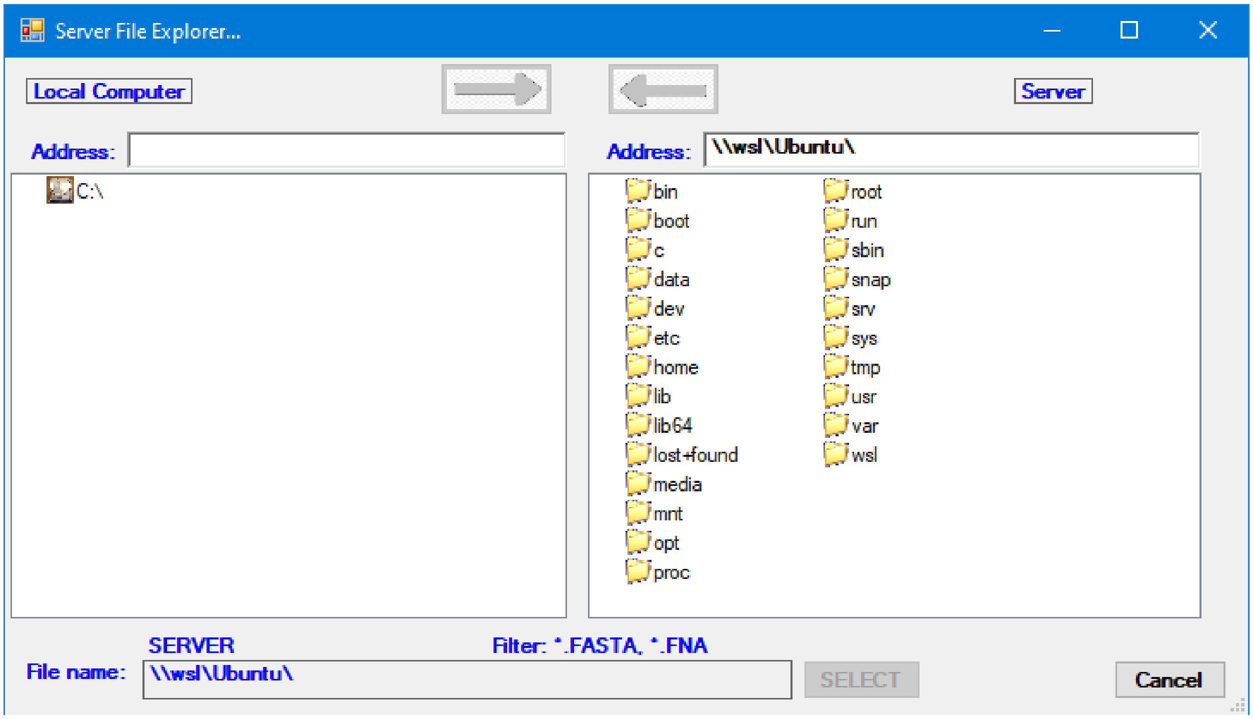
References to the C: and E: drive of WIMMER are represented as /mnt/c and /mnt/e respectively in Linux. For example, a file in the Temp folder on the C: drive might be C:\Temp\stats.txt, but in Linux would be referenced as /mnt/c/Temp/stats.txt. In the Linux reference, upper-case and lower-case is important. Also note the difference in the backslash in the Windows file system and the forward-slash in the Linux file system.

BioSeqDB hides most of these details and translates folder references automatically in the background but understanding these differences between the file systems is important. **A key rule to keep in mind is that no path or file name may contain white space (like a space character)**.

In BioSeqDB, there is also a distinction between files and folders stored on the server and those stored on the client. For example, the output results from running an analysis in BioSeqDB may be stored on either the server or the client computer. To distinguish which file system is intended, the path of the file or folder is prefixed with either a '[S]' if on the server, or a '[L]' if on the local computer.

Navigating file systems

Since the user needs to have detailed access to the file system on the server, BioSeqDB has a unique file and folder explorer to navigate both the local computer and the server file system. The local computer is described on the left side of the dialog and the server is described on the right side. This layout should be familiar to users of Globus, FTP or various other remote connection tools.



Address: \\wsl\Ubuntu\

SERVER

Filter: *.FASTA, *.FNA

File name: \\wsl\Ubuntu\

Double-click on any folder name to drill down to the next level. At times the client application is looking for a file and other times for a folder. The 'SELECT' command is disabled until a valid selection of a file or folder is made. When the 'SELECT' command is enabled and clicked, the dialog is closed with the currently selected path returned to the application.

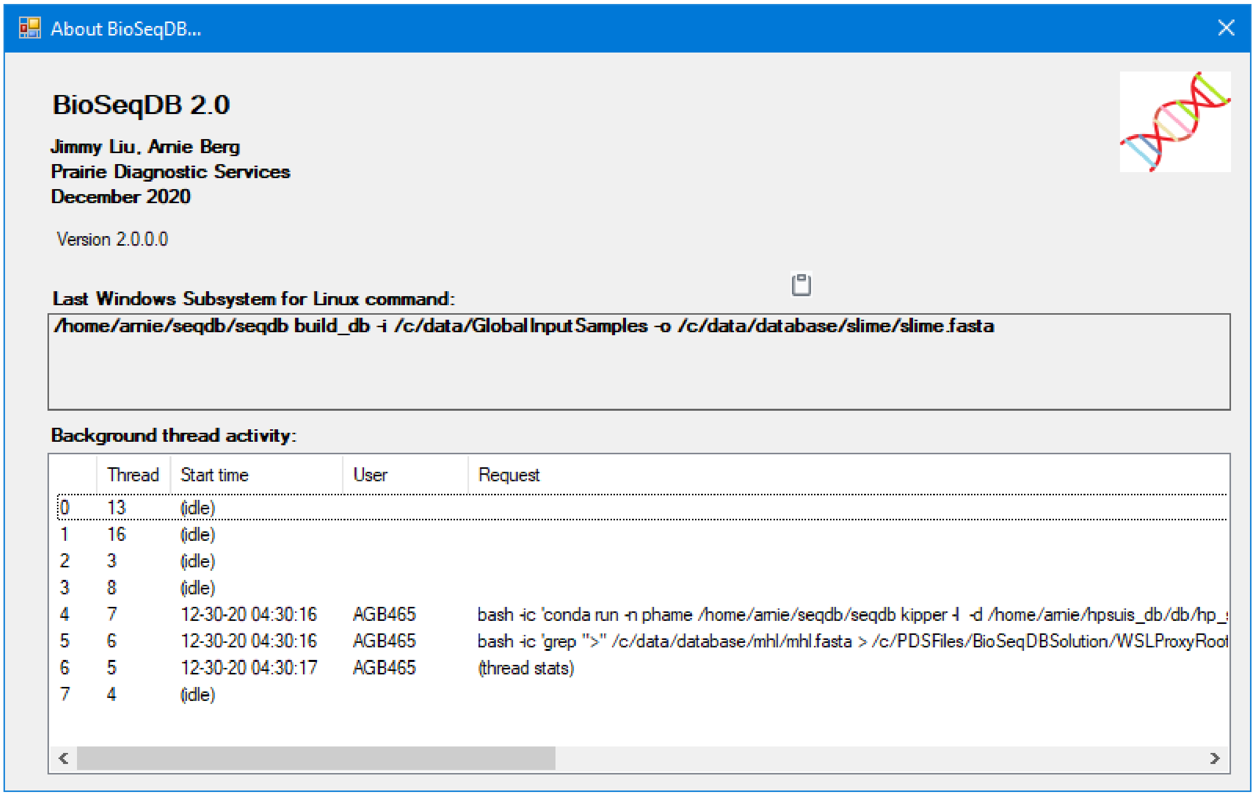
Click on the 'Local Computer' label or 'Server' label at the top of the dialog to reset the explorer back to the contents of the root folder.

Note that this explorer is capable of drilling deep into the WSL Linux file structure as easily as the Windows file structure. Also, if any mapped drives exist on either local computer or the server, that drive shows up as well.

There are two arrows at the top of the dialog. They become enabled any time that a file transfer from the local computer to the server (or vice versa) is valid. This allows transfer of data between computers if needed. Be aware though that if you specify a file as input on your local computer, BioSeqDB automatically and seamlessly looks after transferring that file to the server for processing. The same applies to specifying output results on your local computer.

About dialog

The About dialog for BioSeqDB 2.0 can be opened by clicking on the DNA double-helix symbol on the main dialog. This dialog identifies the version of BioSeqDB along with development credits, but there are a couple of items of added value.



BioSeqDB 2.0

Jimmy Liu, Arnie Berg
Prairie Diagnostic Services
December 2020

Version 2.0.0.0

Last Windows Subsystem for Linux command:

```
/home/arnie/seqdb/seqdb build_db -i /c/data/GlobalInputSamples -o /c/data/database/slime/slime.fasta
```

Background thread activity:

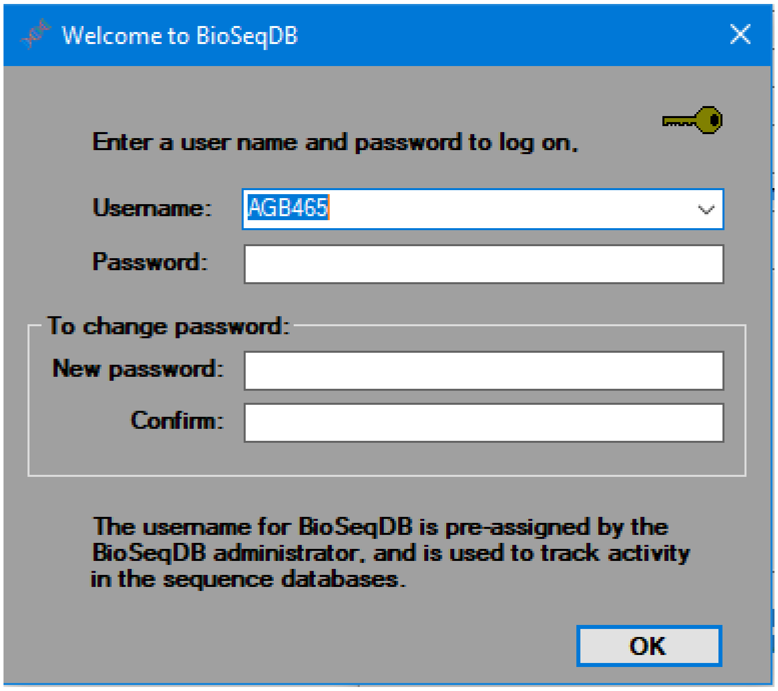
	Thread	Start time	User	Request
0	13	(idle)		
1	16	(idle)		
2	3	(idle)		
3	8	(idle)		
4	7	12-30-20 04:30:16	AGB465	bash -ic 'conda run -n phame /home/arnie/seqdb/seqdb kipper -l -d /home/arnie/hpsuis_db/db/hp_
5	6	12-30-20 04:30:16	AGB465	bash -ic 'grep ">" /c/data/database/mhl/mhl.fasta > /c/PDSFiles/BioSeqDBSolution/WSLProxyRoot
6	5	12-30-20 04:30:17	AGB465	(thread stats)
7	4	(idle)		

The ‘Last Windows Subsystem for Linux command’ lists the last command that was issued to WSL on behalf of the client. For testing and debugging purposes, this command may be copied to the clipboard by clicking on the clipboard icon.

The other bonus item is a snapshot of the background thread activity. All requests to the Linux system are handled by a component on the server called WSLProxy. At this time there can be up to eight simultaneous active threads. The snapshot provides a report of what request is running on each thread, along with the requesting user and the start time. The list can be refreshed by clicking on the DNA double-helix symbol in the top right-hand corner of the About dialog.

Login dialog

BioSeqDB authenticates each user of the application (for administrators, see the section below on ‘User Management’). The username is selected from a dropdown list and the password is entered to be validated. If the password is valid, it can also be changed by supplying a new password in the ‘New password’ and ‘Confirm’ fields.



Enter a user name and password to log on.

Username: AGB465

Password:

To change password:

New password:

Confirm:

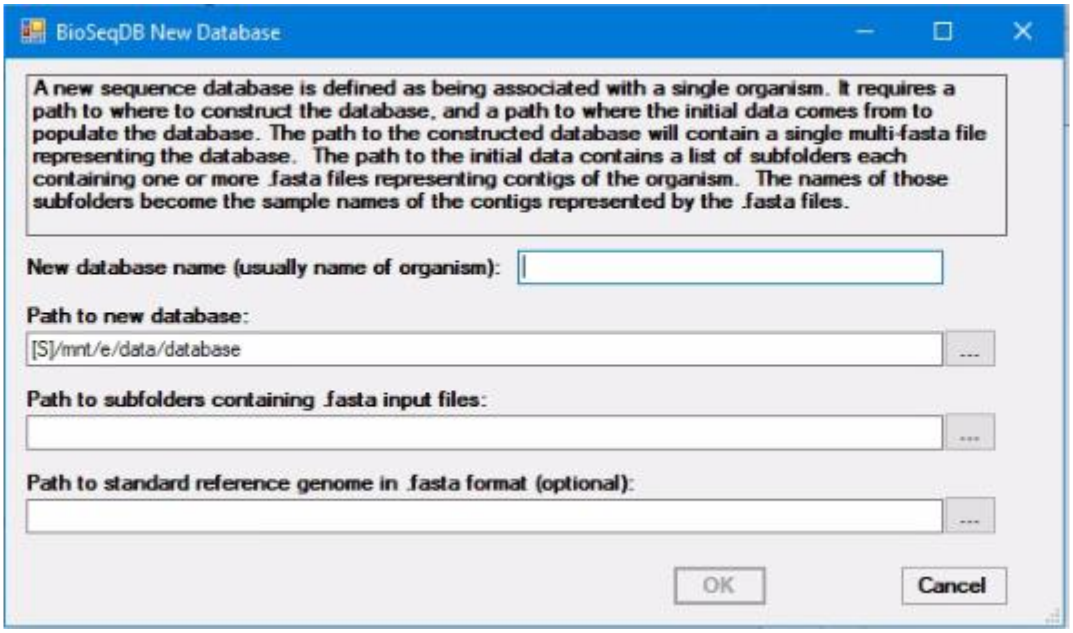
The username for BioSeqDB is pre-assigned by the BioSeqDB administrator, and is used to track activity in the sequence databases.

Functions

The menu strip of BioSeqDB has the list of functions available, including Assemble, Insert, Extract, and Remove. From the main dialog, command buttons are available to Delete, Backup and Restore the selected database. From the '—Select analysis –' dropdown, analysis functions are selected to perform specialized analyses, such as BBMap, Build tree, Influenza A pipeline, Kraken2, Quast, Search and VFabricate. These analyses are described separately below.

A new database may be created by selecting '<new…>' from the 'Sequence database:' dropdown and supplying the sequence database setup information.

0. New database



A new sequence database is defined as being associated with a single organism. It requires a path to where to construct the database, and a path to where the initial data comes from to populate the database. The path to the constructed database will contain a single multi-fasta file representing the database. The path to the initial data contains a list of subfolders each containing one or more .fasta files representing contigs of the organism. The names of those subfolders become the sample names of the contigs represented by the .fasta files.

New database name (usually name of organism):

Path to new database:
[S]/mnt/e/data/database

Path to subfolders containing .fasta input files:

Path to standard reference genome in .fasta format (optional):

A new database requires a database name, which is usually the name of the organism or its abbreviation. The second parameter is the path to the new database. By convention, sequence databases are stored in the E:\data\database folder. The third parameter is the path to the input sequence(s) representing the initial content of the database. This data may be stored anywhere, but it is important to understand the structure of this data.

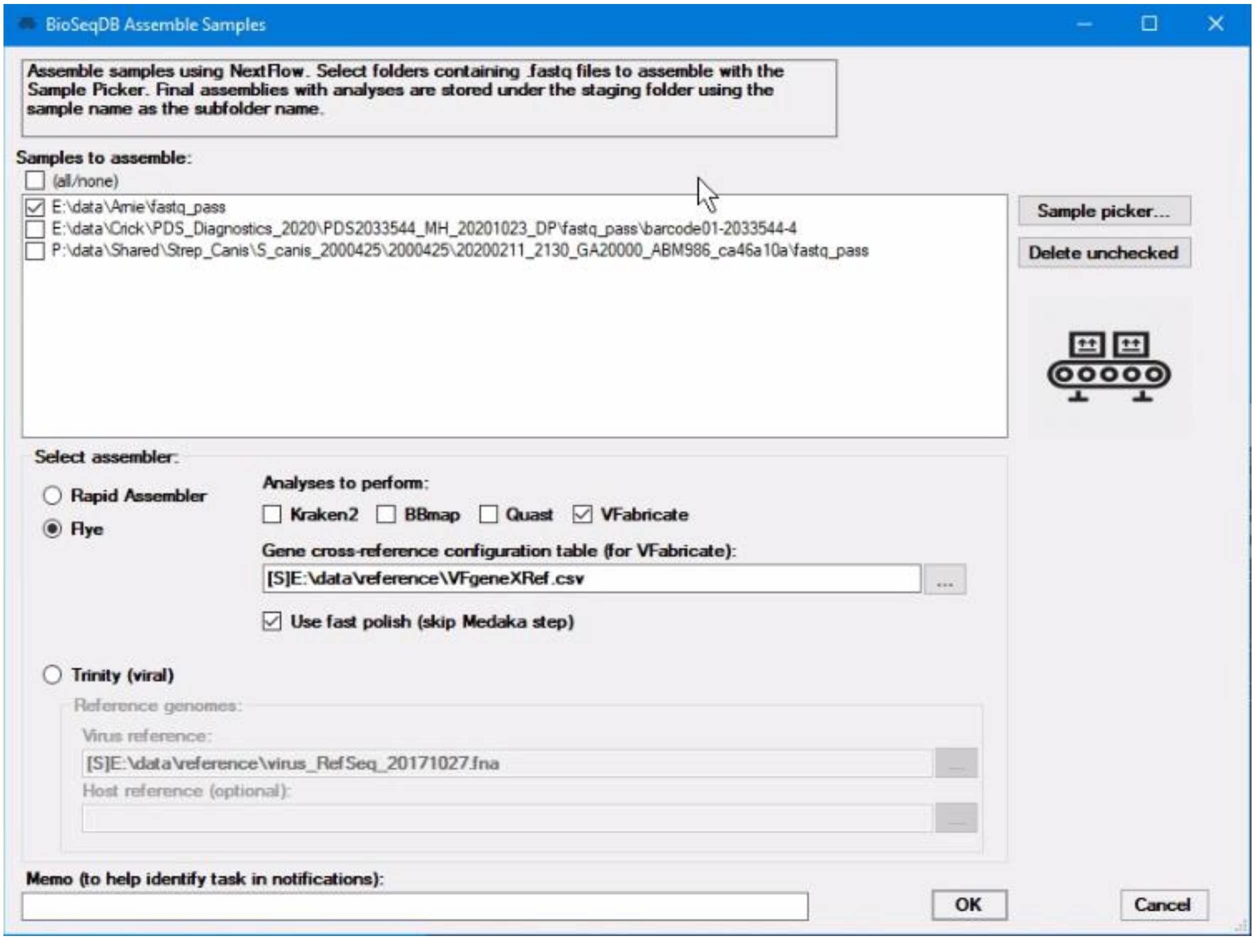
Specify the path containing one or more subfolders, where each subfolder contains one or more .fasta files representing contigs of the organism. The names of the subfolders are the sample IDs that are used to name the sample in the new database. Unless this structure is set up ahead of creating the new database, the setup of the new database will not succeed.

Optionally a standard reference sequence genome may be specified to be associated with the sequence database. This reference genome is useful when running the Build tree function for this database.

1. Assemble

To create new data to add to a sequence database, raw sequence data must be assembled to create contig data in .fasta files. The Assemble function performs this step.

The Assemble function can assemble multiple sets of sequence data at one time. In fact, it is desirable to assemble multiple sequence data at once to take advantage of the multi-tasking capability of WIMMER. Use the Sample Picker to select the folder containing the .fastq files to include in the Assemble step. Only samples that are checked in the Samples list are included in the Assemble function. Any unchecked samples can be deleted with the 'Delete unchecked' command button. The sample name is derived from the immediate parent folder of the .fastq data. For example, in the selected sample below, the sample name is 'barcode01-2033544-4'.



BioSeqDB Assemble Samples

Assemble samples using NextFlow. Select folders containing .fastq files to assemble with the Sample Picker. Final assemblies with analyses are stored under the staging folder using the sample name as the subfolder name.

Samples to assemble:

□ (all/none)

☑ E:\data\Arnie\fastq_pass

□ E:\data\Crick\PDS_Diagnostics_2020\PDS2033544_MH_20201023_DP\fastq_pass\barcode01-2033544-4

□ P:\data\Shared\Strep_Canis\S_canis_2000425\2000425\20200211_2130_GA20000_ABM986_ca46a10a\fastq_pass

Select assembler:

○ Rapid Assembler

● Flye

Analyses to perform:

□ Kraken2 □ BBmap □ Quast ☑ VFabricate

Gene cross-reference configuration table (for VFabricate):

[S]E:\data\reference\VFgeneXRef.csv

☑ Use fast polish (skip Medaka step)

○ Trinity (viral)

Reference genomes:

Virus reference:

[S]E:\data\reference\virus_RefSeq_20171027.fna

Host reference (optional):

Memo (to help identify task in notifications):

Two bacterial assemblers are available, either Flye or Rapid Assembler. Although they produce similar results, sometimes one will fail to successfully complete the assembly and the other will succeed. There is also an optional Medaka step which further polishes the data but takes an increased amount of time. Any of four analyses to perform can be selected, Kraken2, BBmap

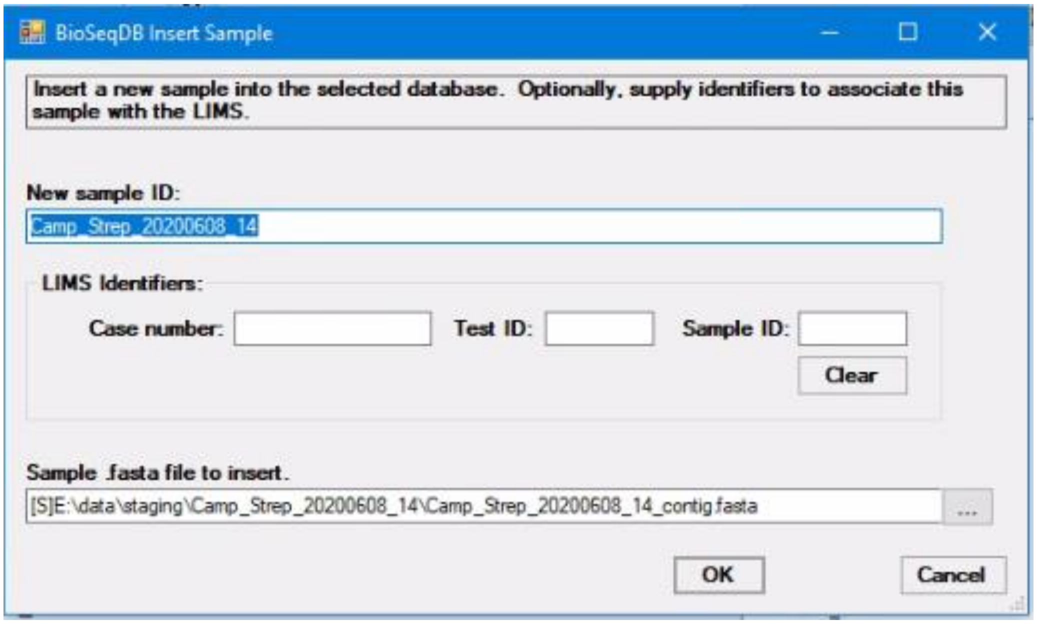
Quast and VFabricate. If VFabricate is selected, the gene cross-reference configuration table can also be defined and accessed for editing.

A viral assembler based on Trinity is also available. The assembly is based on the reference genome for the virus. If the data contain host sequences, a host reference genome may optionally be specified to remove the host DNA data. Because this assembly uses porechop to demultiplex and polish the data, this assembly can run for hours, depending on the amount of data. This assembly workflow requires that the viral data be barcoded.

The results of the Assemble function, if successful, are stored in the E:\data\staging folder, with a subfolder created named after the sample name of the data assembled. A .fasta contig file is created that can then be inserted into a sequence database.

The Assemble function is one of the functions that can be quite time-consuming. As such, it is scheduled to run in the background and its status is referred to 'Pending' until it completes and becomes 'Ready'. The 'Pending Notifications' dialog provides the ability to manage this process. The details about scheduled functions are provided below in 'Running in the background'. The optional 'Memo' field at the bottom of the Assemble dialog can be used to associate details relevant to the assembly in order to better track the task in the background.

2. Insert



Insert a new sample into the selected database. Optionally, supply identifiers to associate this sample with the LIMS.

New sample ID: Camp_Strep_20200608_14

LIMS Identifiers:

Case number: Test ID: Sample ID:

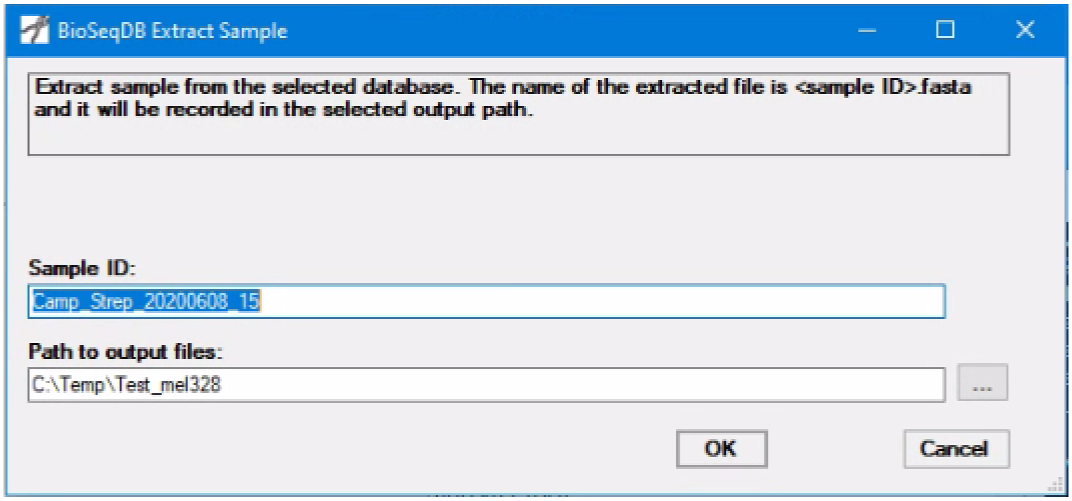
Sample .fasta file to insert.: [S]E:\data\staging\Camp_Strep_20200608_14\Camp_Strep_20200608_14_contig.fasta

The Insert function inserts the contents of a .fasta contig file into the current database. You are prompted to create a new sample ID and select the .fasta file containing the contig(s).

If the sample ID you enter already exists in the database, you are prompted as to whether you want to replace the existing data associated with the sample ID in the database.

Optionally, to connect the sequence sample with the sample in the LIMS, a case number, LIMS test ID and LIMS sample ID may be specified. These LIMS identifiers may also be edited by clicking on the 'LIMS edit sample identifiers' command button on the main dialog to open a dialog to edit values for any sequence sample in the currently selected sequence database. For more details, see the section below entitled 'Editing LIMS identifiers'.

3. **Extract**



BioSeqDB Extract Sample

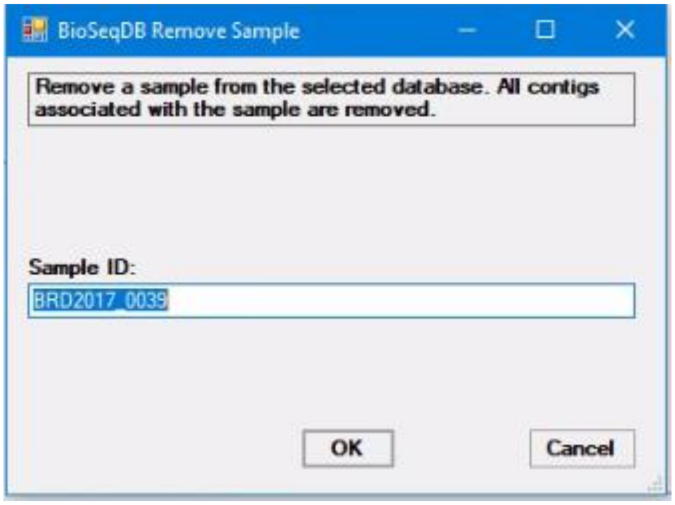
Extract sample from the selected database. The name of the extracted file is <sample ID>.fasta and it will be recorded in the selected output path.

Sample ID: Camp_Strep_20200608_15

Path to output files: C:\Temp\Test_mel328

The Extract function is used to extract a sequence from the currently selected database. You are prompted for a Sample ID and a path where the extracted sequence can be recorded. Since the sample is selected from the currently selected database, it is enough just to select the sample ID from the sample ID list on the main dialog.

4. **Remove**



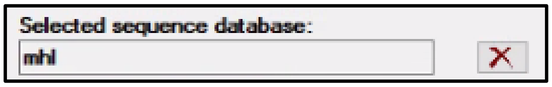
BioSeqDB Remove Sample

Remove a sample from the selected database. All contigs associated with the sample are removed.

Sample ID: BRD2017_0039

As with the Extract function, the Remove function uses the specified Sample ID, either entered from the keyboard or selected from the sample ID list. The specified Sample ID is removed from the sequence database if it exists.

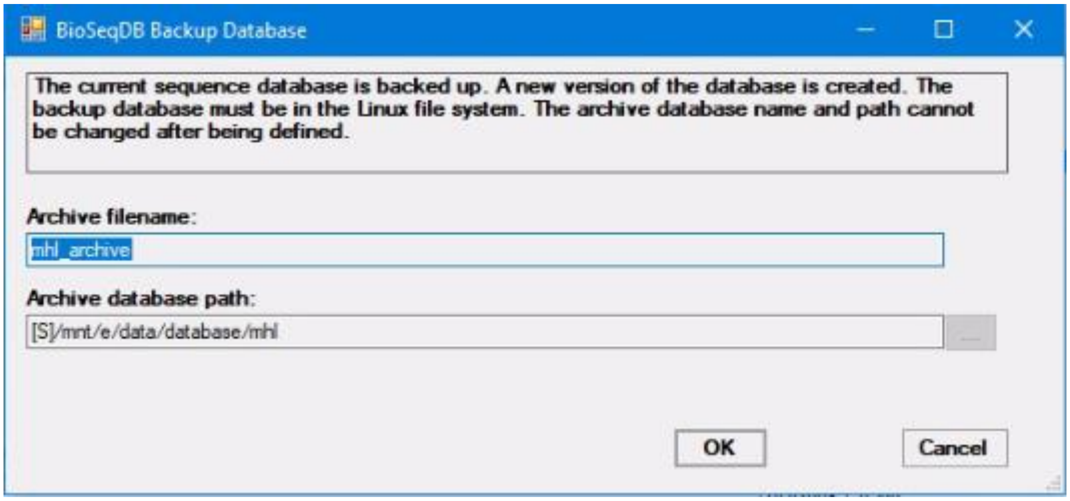
5. Delete database



Selected sequence database: mhl

A sequence database may be deleted by clicking on the red 'X' to the right of the selected sequence database name. A confirmation prompt appears to verify that the intention is to actually delete the database.

6. Backup database



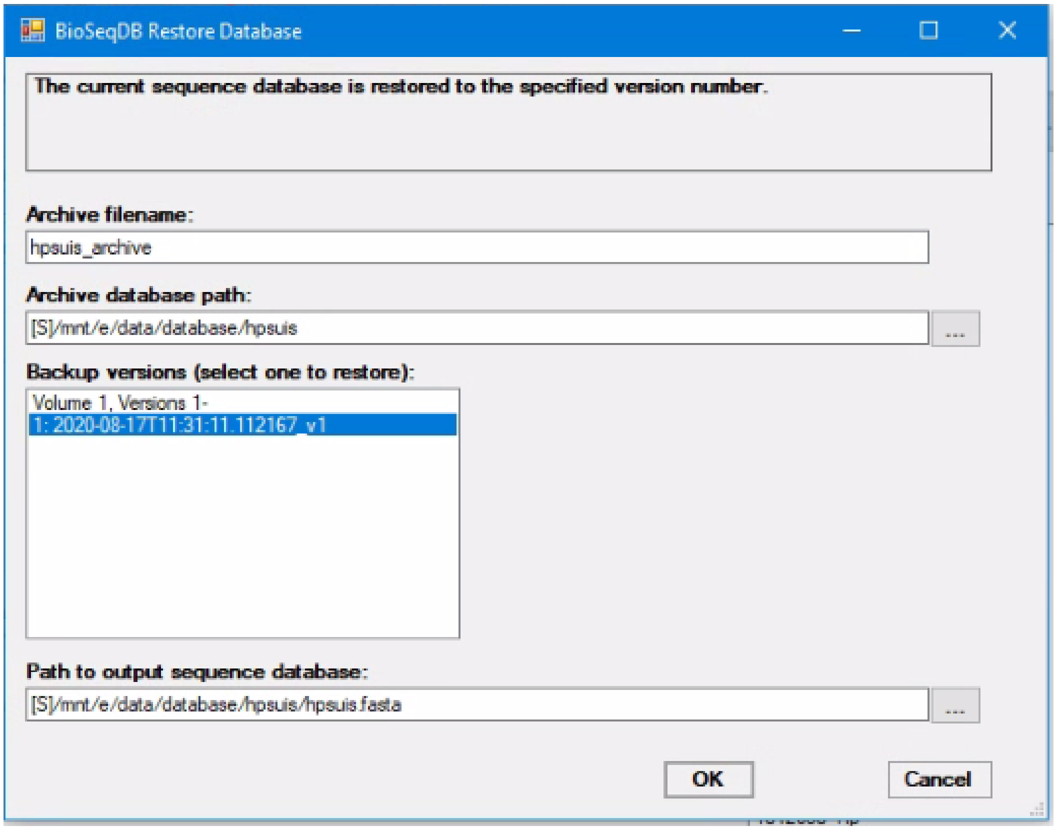
The current sequence database is backed up. A new version of the database is created. The backup database must be in the Linux file system. The archive database name and path cannot be changed after being defined.

Archive filename: mhl_archive

Archive database path: [S]/mnt/e/data/database/mhl

BioSeqDB uses a software tool called Kipper to create an incremental backup of the currently selected database. Each time the Backup function is invoked, only the incremental *changes* to the database are recorded, and a new date/time stamped version of the database is created. The versions are listed on the left of the main dialog. This approach results in a very space-efficient means of creating multiple versions of the database. It is recommended that a backup be performed whenever a significant number of changes to the database have taken place.

7. Restore database



BioSeqDB Restore Database

The current sequence database is restored to the specified version number.

Archive filename: hpsuis_archive

Archive database path: [S]/mnt/e/data/database/hpsuis

Backup versions (select one to restore):

Volume 1, Versions 1-
1: 2020-08-17T11:31:11.112167_v1

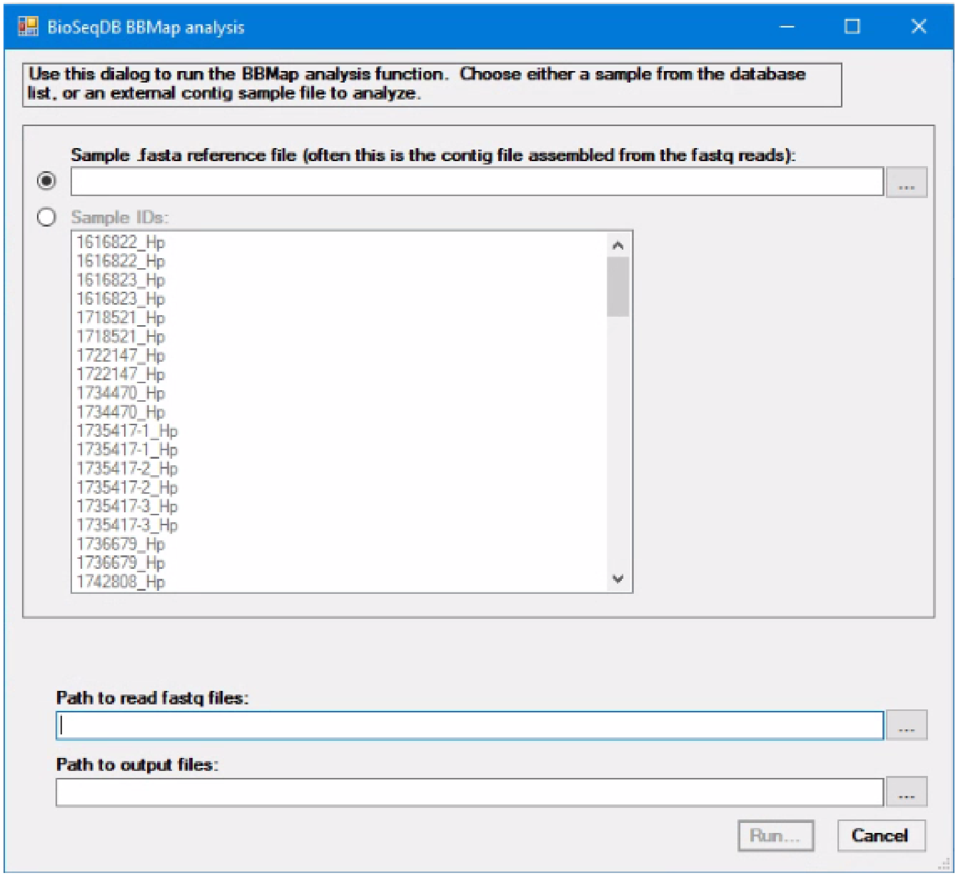
Path to output sequence database: [S]/mnt/e/data/database/hpsuis/hpsuis.fasta

The Restore function can restore the currently selected database to any point in time from the list of backup database versions available. Select the version to restore by clicking on the desired version in the 'Backup versions' list. The currently selected database is replaced by the restored database. If you are restoring an older version of the database but want to later return to the currently selected database, be sure to take a backup first so that later you can restore the current version.

Analysis Functions

Various analysis functions are available which run either in conjunction with data from the sequence databases or with external data. The functions all run as background tasks when activated. Each of the analysis dialogs has an optional memo field to help identify the function when it runs in the background.

0. BBMap



Use this dialog to run the BBMap analysis function. Choose either a sample from the database list, or an external contig sample file to analyze.

● Sample .fasta reference file (often this is the contig file assembled from the fastq reads):

○ Sample IDs:

1616822_Hp
1616822_Hp
1616823_Hp
1616823_Hp
1718521_Hp
1718521_Hp
1722147_Hp
1722147_Hp
1734470_Hp
1734470_Hp
1735417-1_Hp
1735417-1_Hp
1735417-2_Hp
1735417-2_Hp
1735417-3_Hp
1735417-3_Hp
1736679_Hp
1736679_Hp
1742808_Hp

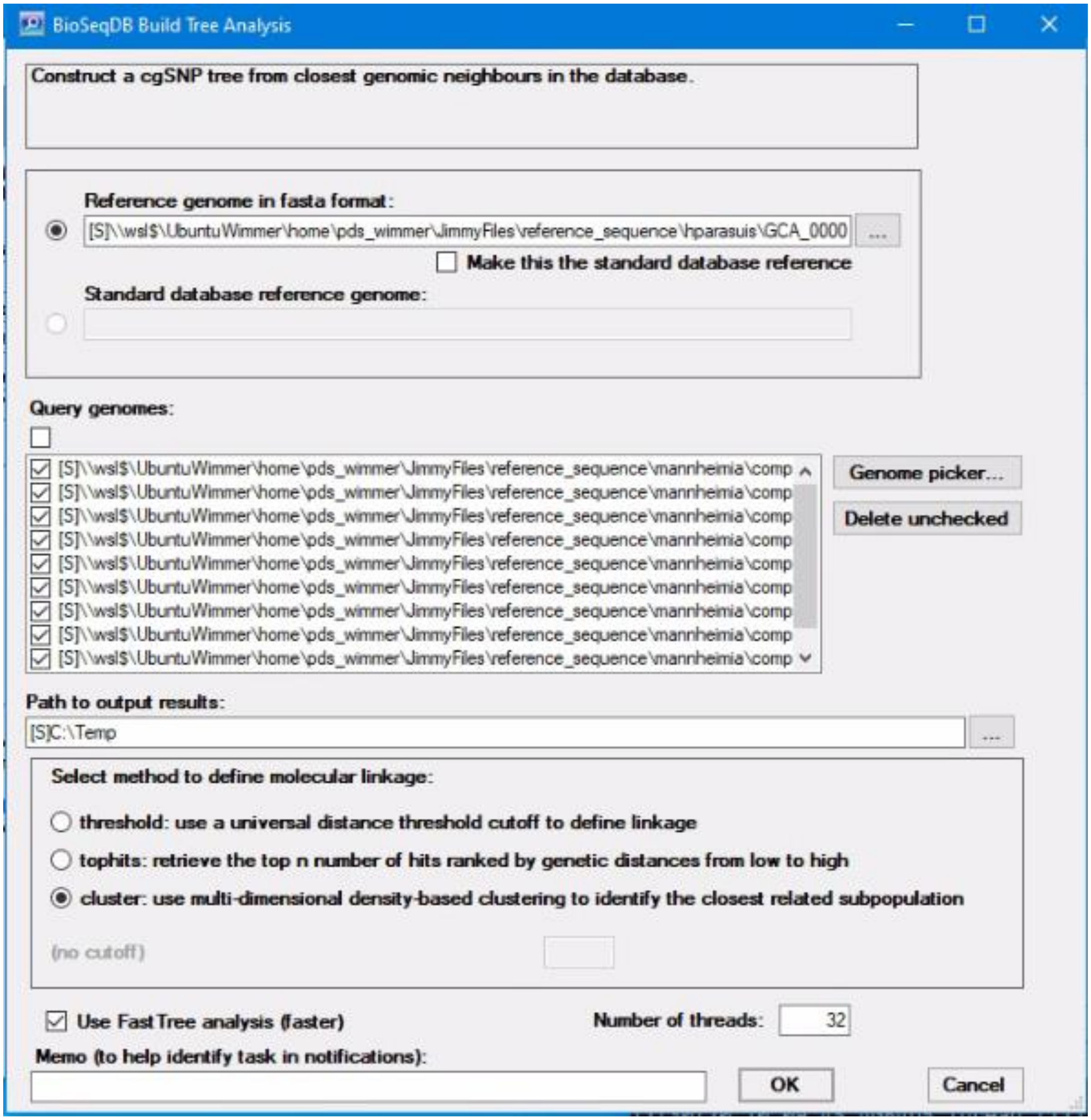
Path to read fastq files:

Path to output files:

The BBMap analysis takes as input a reference .fasta file from either an external contig file or a sample from within the selected database. Also specify a path to the .fastq files to match against the reference, as well as a path to where the output results are to be stored.

1. Build tree

The Build tree function is one of the key phylogenetic analysis tools available. The purpose of this function is not limited to just the phylogenetic reconstruction of a set of query genomes, but it also searches the query genomes against database sequences to identify historical strains that demonstrate molecular linkage to query genomes. This functionality enables rapid identification of sequences which likely share lineage origins with query sequences, enabling one to infer epidemiological origins and unknown phenotypic characteristics. To run the phylogenetic analysis, you must specify a reference genome for SNP typing, a set of query genomes, an output path for the results, and a method to define molecular linkage/close relatedness. More information below.



Construct a cgSNP tree from closest genomic neighbours in the database.

● Reference genome in fasta format: [S]\\wsl$\UbuntuWimmer\home\pds_wimmer\JimmyFiles\reference_sequence\hparasuis\GCA_0000

□ Make this the standard database reference

○ Standard database reference genome:

Query genomes:

□

☑ [S]\\wsl$\UbuntuWimmer\home\pds_wimmer\JimmyFiles\reference_sequence\mannheimia\comp
☑ [S]\\wsl$\UbuntuWimmer\home\pds_wimmer\JimmyFiles\reference_sequence\mannheimia\comp
☑ [S]\\wsl$\UbuntuWimmer\home\pds_wimmer\JimmyFiles\reference_sequence\mannheimia\comp
☑ [S]\\wsl$\UbuntuWimmer\home\pds_wimmer\JimmyFiles\reference_sequence\mannheimia\comp
☑ [S]\\wsl$\UbuntuWimmer\home\pds_wimmer\JimmyFiles\reference_sequence\mannheimia\comp
☑ [S]\\wsl$\UbuntuWimmer\home\pds_wimmer\JimmyFiles\reference_sequence\mannheimia\comp
☑ [S]\\wsl$\UbuntuWimmer\home\pds_wimmer\JimmyFiles\reference_sequence\mannheimia\comp
☑ [S]\\wsl$\UbuntuWimmer\home\pds_wimmer\JimmyFiles\reference_sequence\mannheimia\comp
☑ [S]\\wsl$\UbuntuWimmer\home\pds_wimmer\JimmyFiles\reference_sequence\mannheimia\comp

Path to output results: [S]C:\Temp

Select method to define molecular linkage:

○ threshold: use a universal distance threshold cutoff to define linkage
○ tophits: retrieve the top n number of hits ranked by genetic distances from low to high
● cluster: use multi-dimensional density-based clustering to identify the closest related subpopulation

(no cutoff)

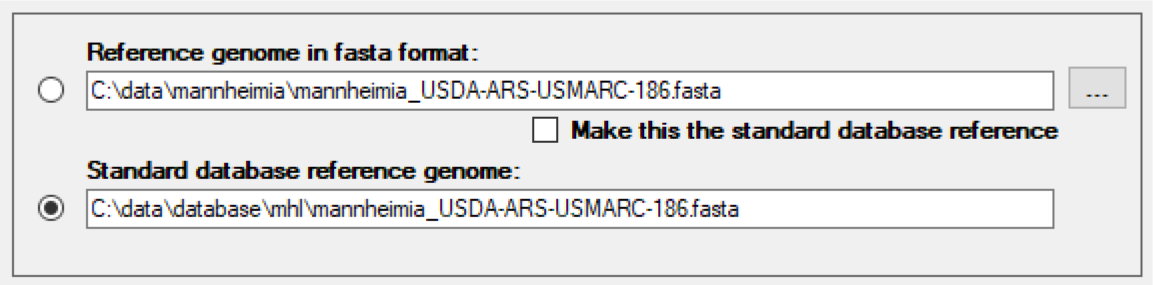
☑ Use FastTree analysis (faster)

Number of threads: 32

Memo (to help identify task in notifications):

Either of two references genomes can be used, one that is a standard reference associated with the currently selected database, and the other chosen from anywhere in the file system.

Use the Memo field to enter identifying information to facilitate tracking the background task in the Notifications dialog.



Reference genome in fasta format:

○ C:\data\mannheimia\mannheimia_USDA-ARS-USMARC-186.fasta

□ Make this the standard database reference

Standard database reference genome:

● C:\data\database\mhl\mannheimia_USDA-ARS-USMARC-186.fasta

Use the radio buttons to select which reference genome will be used.

The only way to record a standard reference genome is to select it from the file system and check the checkbox that says, ‘Make this the standard database reference’. When the Build tree function runs, the reference genome from the file system is copied to the currently selected database path as the standard reference genome.

Use the ‘Genome picker’ to select the series of query genomes. All checked genomes are included in the analysis. To remove a genome from the list, uncheck it and select ‘Delete unchecked’. This removes ALL genomes that are unchecked in the list.

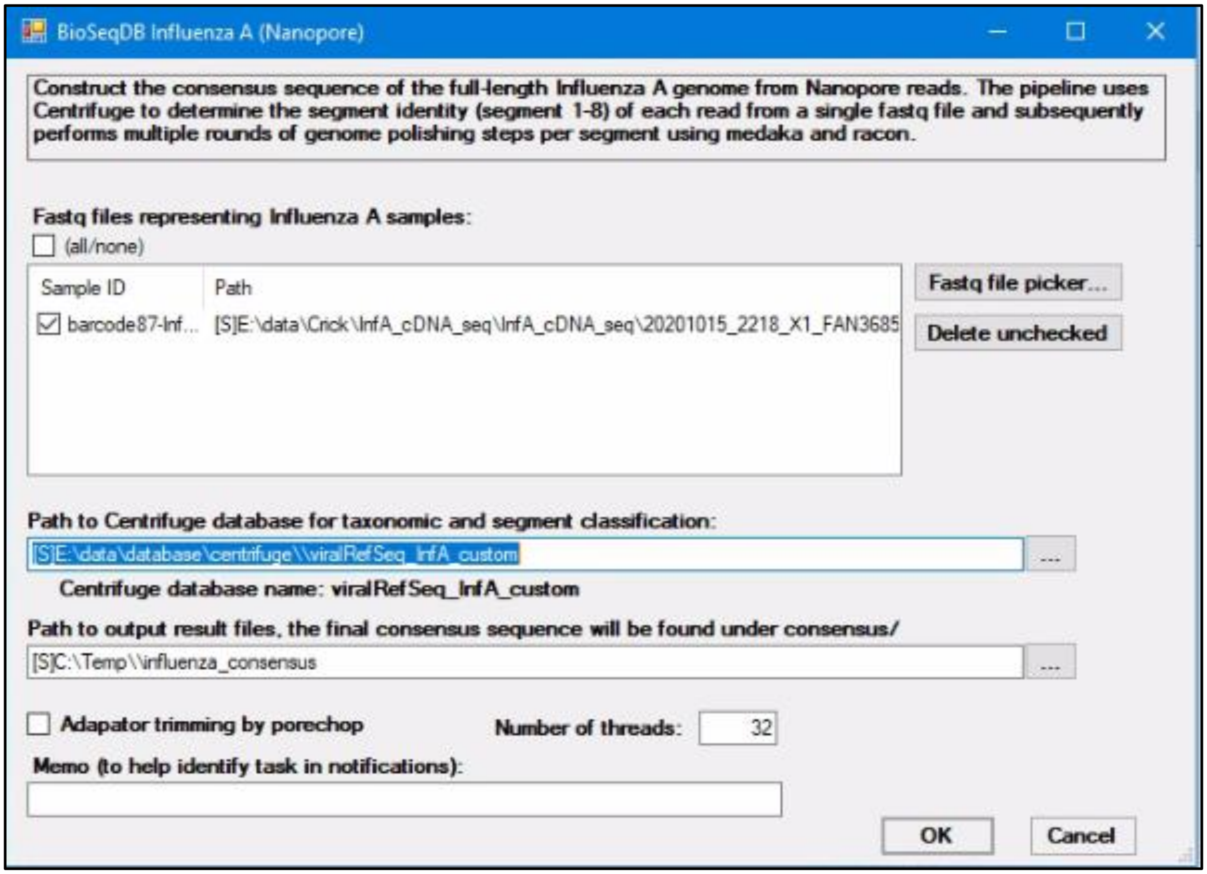
To identify close genomic neighbors in the database, a method to define molecular linkage must be selected. There are three different methods available:

1. **threshold** – set a universal cutoff value in which sequences below the given distance threshold are defined as close genomic neighbors
2. **tophits** – the top *n* number of database sequences ranked by genetic distances from low to high are defined as close genomic neighbors
3. **cluster** – a non-parametric method that uses density-based clustering to dynamically characterize subpopulations based on the genetic distances between the query and database sequences. The subpopulation with the shortest distance to the query is selected as close genomic neighbors.

There are other optional parameters that may be specified. A checkbox to specify ‘Use Fast Tree analysis’ results in an analysis that runs much faster but is slightly less accurate. Another option is to adjust the distance threshold values for the linkage methods. This only applies to \`threshold\` and \`tophits\` linkage methods, as the \`cluster\` method is non-parametric.

The output of the Build tree function is two files, tree.nwk and metadata_microreact.csv. The tree.nwk file consists of the tree description in Newick format, and the metadata_microreact.csv file that consists of a list of sample IDs mapped to the status attribute. The status attribute indicates whether a given sample in the tree is a query or a local database sequence. You can append more sample attributes to this file such as AMR profiles or serotypes to highlight certain trends in the phylogentic tree. Note that the metadata_microreact.csv is only designed to annotate trees visualized on the Microreact Internet platform. Upon analysis completion, the Build tree function automatically invokes the Dendroscope program to display the tree graphically, but the files may also be manually uploaded to the https://microreact.org/upload website for an alternative method of displaying the tree.

2. Influenza A pipeline



BioSeqDB Influenza A (Nanopore)

Construct the consensus sequence of the full-length Influenza A genome from Nanopore reads. The pipeline uses Centrifuge to determine the segment identity (segment 1-8) of each read from a single fastq file and subsequently performs multiple rounds of genome polishing steps per segment using medaka and racon.

Fastq files representing Influenza A samples:

☐ (all/none)

Sample ID	Path
☑ barcode87-Inf...	[S]E:\data\Crick\InfA_cDNA_seq\InfA_cDNA_seq\20201015_2218_X1_FAN3685

Path to Centrifuge database for taxonomic and segment classification:

[S]E:\data\database\centrifuge\\viralRefSeq_InfA_custom

Centrifuge database name: viralRefSeq_InfA_custom

Path to output result files, the final consensus sequence will be found under consensus/

[S]C:\Temp\\influenza_consensus

☐ Adaptor trimming by porechop

Number of threads: 32

Memo (to help identify task in notifications):

The purpose of this pipeline is to generate the consensus sequence of all eight genomic segments of Influenza A from raw Nanopore sequences that were amplified using universal PCR primers. As such, the required inputs are .fastq files which can be selected using the “Fastq file picker”. Given that the raw Nanopore reads of each sample are partitioned across multiple fastq files, we must select the **DIRECTORY** that contains the individual fastq files of each sample rather than selecting individual fastq files. A sample ID must be given to keep track of each sample and the name of the selected directory is used as the sample ID as default, but can be changed if necessary.

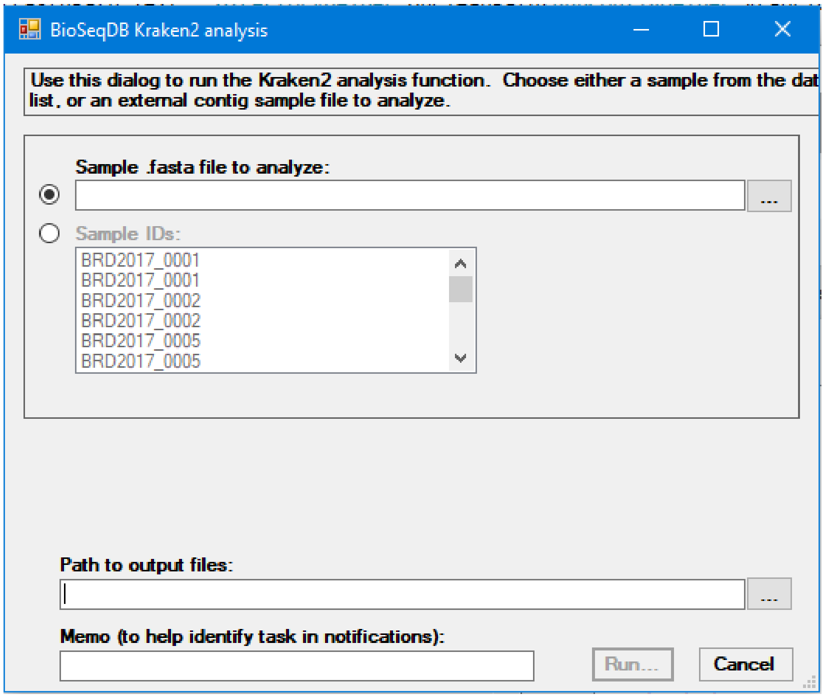
This pipeline uses Centrifuge to perform taxonomic classification and alignment of each read to the genomic segments of Influenza A. The second step is to indicate the **path to the DIRECTORY** that contains the Centrifuge database for taxonomic classification. There is an analysis-ready Centrifuge database in E:\data\database\centrifuge\viralRefSeq_InfA_custom.

The third required input is the path to the output directory where the consensus sequences of each sample will be stored. The output of each sample will be organized into individual subdirectories in the specified output directory. If the output path is on the local computer, the results are copied to that folder on the local machine.

Raw read adaptor and barcode trimming by porechop can be enabled as an optional step; however, it is strongly recommended to avoid having adaptor sequences embedded in the final consensus sequence. The number of threads can also be adjusted to request a higher number of cores to run the analysis.

Use the Memo field to enter identifying information to facilitate tracking the background task in the Notifications dialog.

3. Kraken2



Use this dialog to run the Kraken2 analysis function. Choose either a sample from the dat list, or an external contig sample file to analyze.

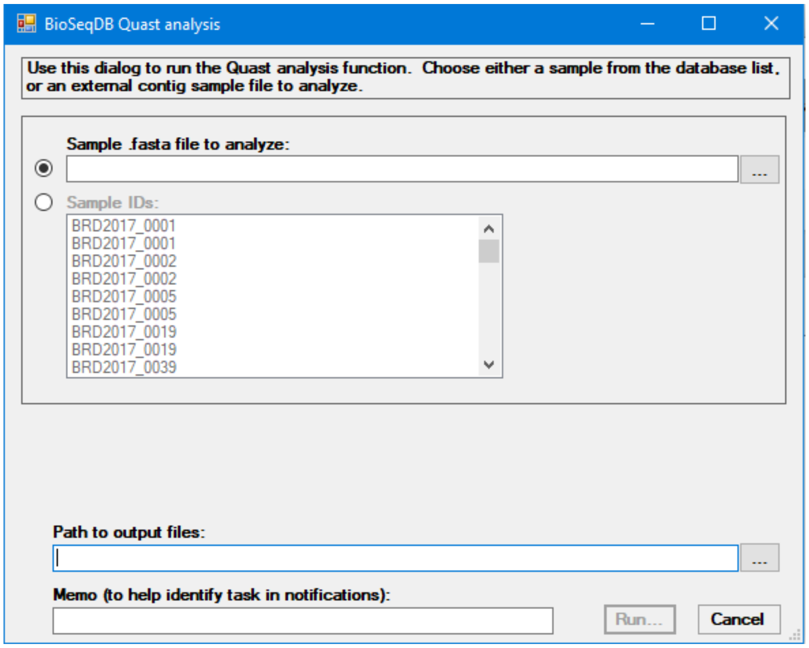
● Sample .fasta file to analyze:

○ Sample IDs:

The Kraken2 analysis takes as input a reference .fasta file from either an external contig file or a sample from within the selected database. Also specify a path to where the Kraken2 output results are to be stored.

Use the Memo field to enter identifying information to facilitate tracking the background task in the Notifications dialog.

4. Quast



Use this dialog to run the Quast analysis function. Choose either a sample from the database list, or an external contig sample file to analyze.

● Sample .fasta file to analyze:

○ Sample IDs:

BRD2017_0001
BRD2017_0001
BRD2017_0002
BRD2017_0002
BRD2017_0005
BRD2017_0005
BRD2017_0019
BRD2017_0019
BRD2017_0039

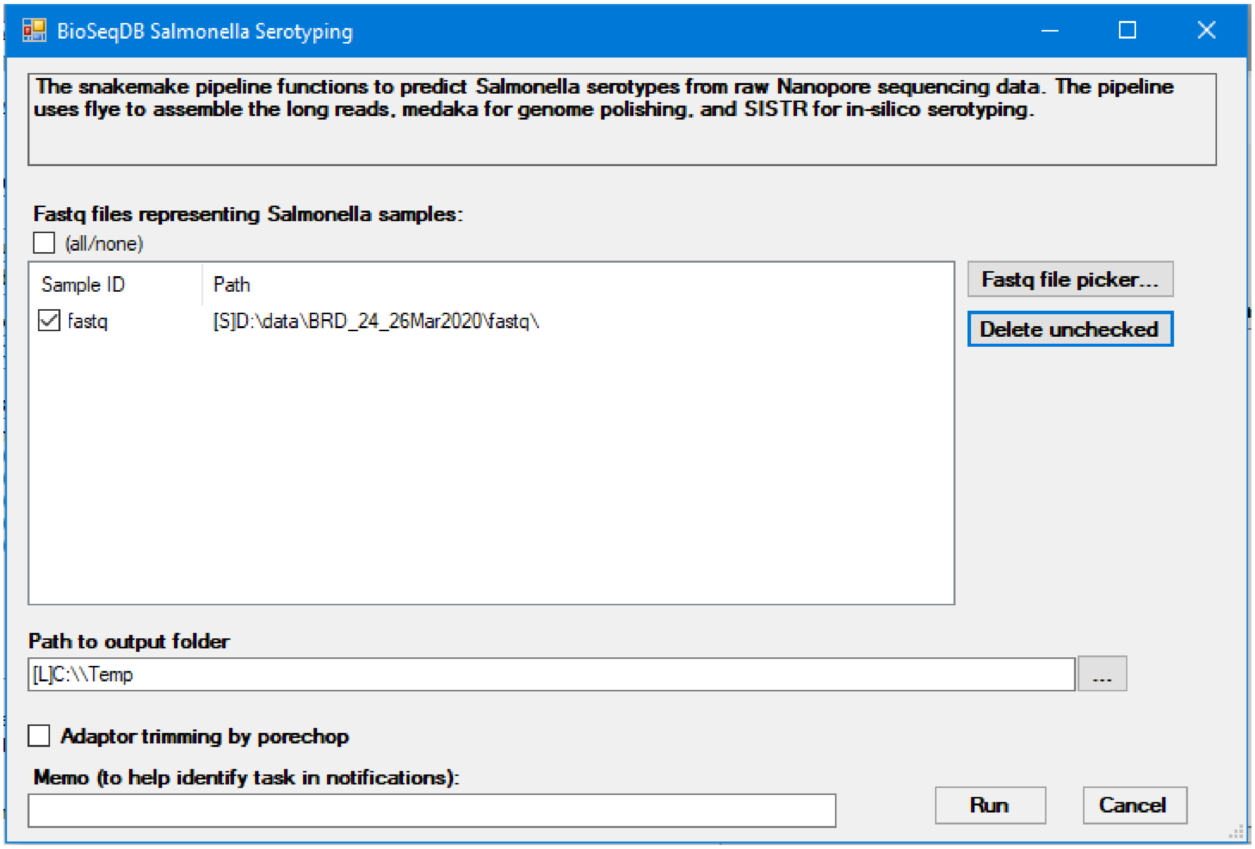
Path to output files:

Memo (to help identify task in notifications):

The Quast analysis takes as input a reference .fasta file from either an external contig file or a sample from within the selected database. Also specify a path to where the Quast output results are to be stored.

Use the Memo field to enter identifying information to facilitate tracking the background task in the Notifications dialog.

5. Salmonella Serotyping



BioSeqDB Salmonella Serotyping

The snakemake pipeline functions to predict Salmonella serotypes from raw Nanopore sequencing data. The pipeline uses flye to assemble the long reads, medaka for genome polishing, and SISTR for in-silico serotyping.

Fastq files representing Salmonella samples:

□ (all/none)

Sample ID	Path
☑ fastq	[S]D:\data\BRD_24_26Mar2020\fastq\

Path to output folder

[L]C:\\Temp

□ Adaptor trimming by porechop

Memo (to help identify task in notifications):

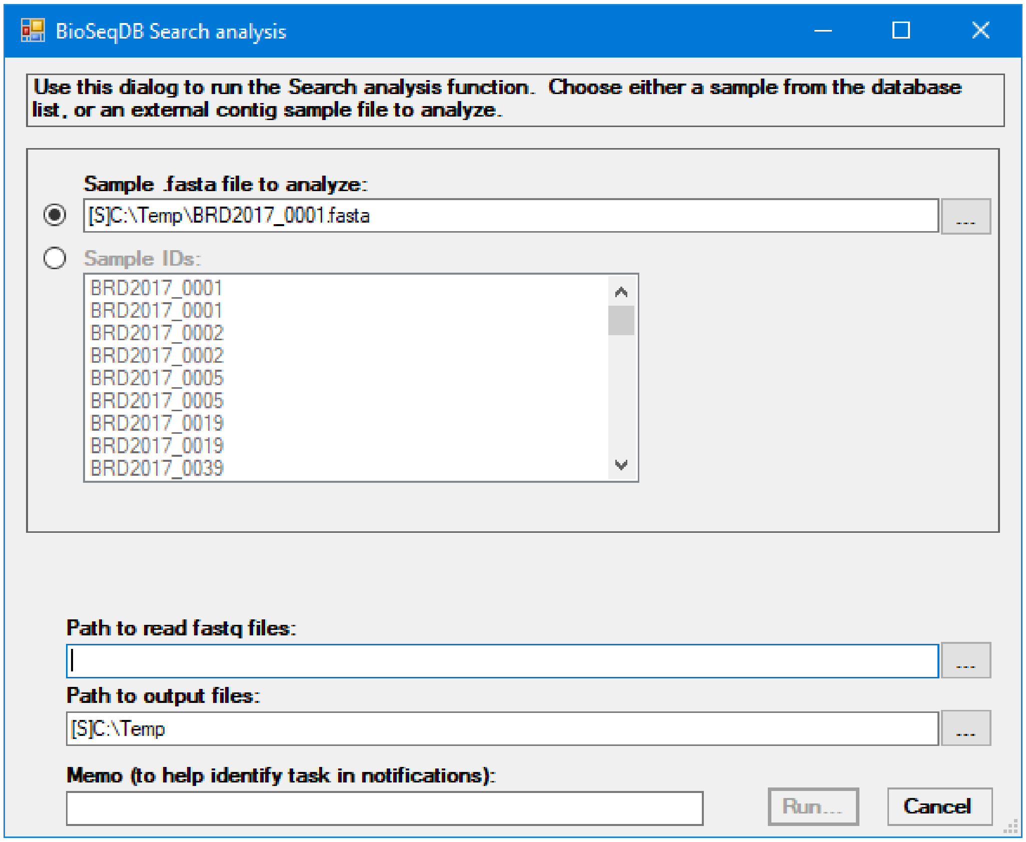
The purpose of this pipeline is to perform in-silico Salmonella serotyping from raw Nanopore sequences. As such, the required inputs are fastq files which can be selected using the “Fastq file picker”. Given that the raw Nanopore reads of each sample are partitioned across multiple fastq files, we must select the **DIRECTORY** that contains the individual fastq files of each sample rather than selecting individual fastq files. A sample ID must be given to keep track of each sample and the name of the selected directory is used as the sample ID as default, but can be changed if necessary.

The second required input is the path to the output directory where the serotyping results will be written to. The serotyping results of all the samples selected in a single analysis run will be aggregated into a single output file named “sistr_res_aggregate{time}.csv”. If the output path is on the local computer, the results are copied to that folder on the local machine and automatically opened.

Raw read adaptor and barcode trimming by porechop can be enabled as an optional step. The number of threads can also be adjusted to request a higher number of cores to run the analysis.

Use the Memo field to enter identifying information to facilitate tracking the background task in the Notifications dialog.

6. Search



Use this dialog to run the Search analysis function. Choose either a sample from the database list, or an external contig sample file to analyze.

● Sample .fasta file to analyze: [S]C:\Temp\BRD2017_0001.fasta

○ Sample IDs:

```
BRD2017_0001
BRD2017_0001
BRD2017_0002
BRD2017_0002
BRD2017_0005
BRD2017_0005
BRD2017_0019
BRD2017_0019
BRD2017_0039
```

Path to read fastq files:

Path to output files: [S]C:\Temp

Memo (to help identify task in notifications):

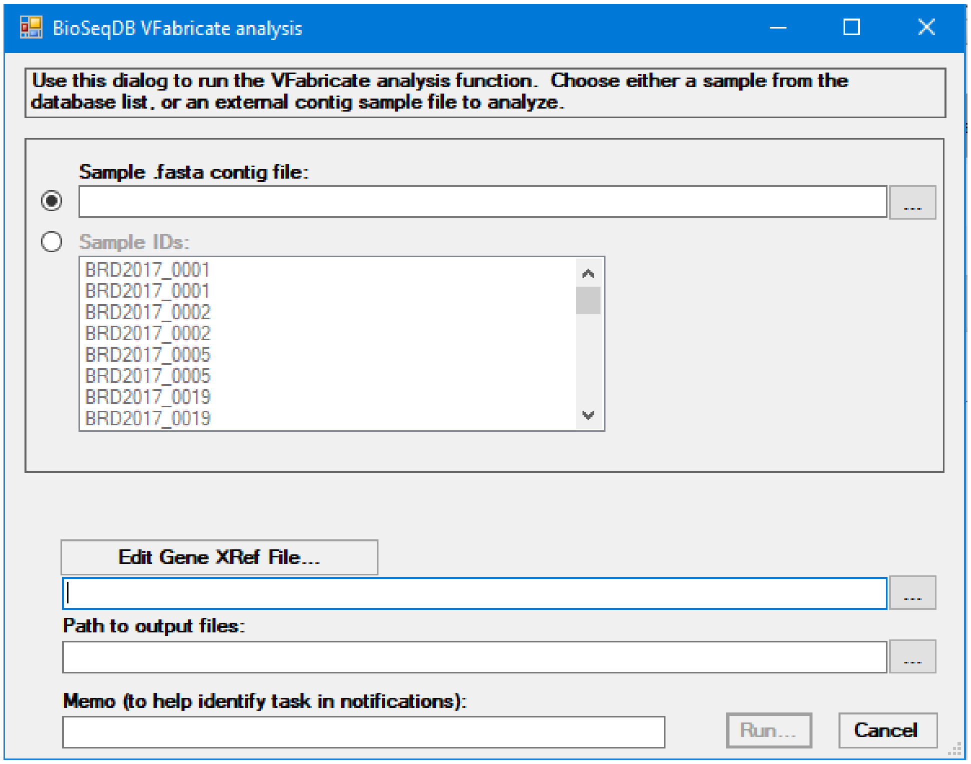
The Search function searches the currently selected database for sequences like a query sequence. You are prompted for a new sample ID that is used to name the output result file, a path to an query sequence in .fasta format, and a path where the output results are stored. If there are no errors, the results are stored in a .txt file and you are prompted to open the file. The result file contains a list of up to 50 sequences that are like the query sequence in descending order of similarity.

Two options are available to qualify the search. The results may be filtered by a maximum distance cutoff, ranging from 0 to 1 with a default of 1. The number of threads may also be specified to speed up the search.

Use the Memo field to enter identifying information to facilitate tracking the background task in the Notifications dialog.

7. VFabricate

The VFabricate analysis uses Abricate to report on the frequency of virus genes in a given sample. The sample comes from either an external contig file or a sample from within the selected database. Also specify a path to where the VFabricate output results are to be stored.



BioSeqDB VFabricate analysis

Use this dialog to run the VFabricate analysis function. Choose either a sample from the database list, or an external contig sample file to analyze.

● Sample .fasta contig file:

○ Sample IDs:

BRD2017_0001
BRD2017_0001
BRD2017_0002
BRD2017_0002
BRD2017_0005
BRD2017_0005
BRD2017_0019
BRD2017_0019

Path to output files:

Memo (to help identify task in notifications):

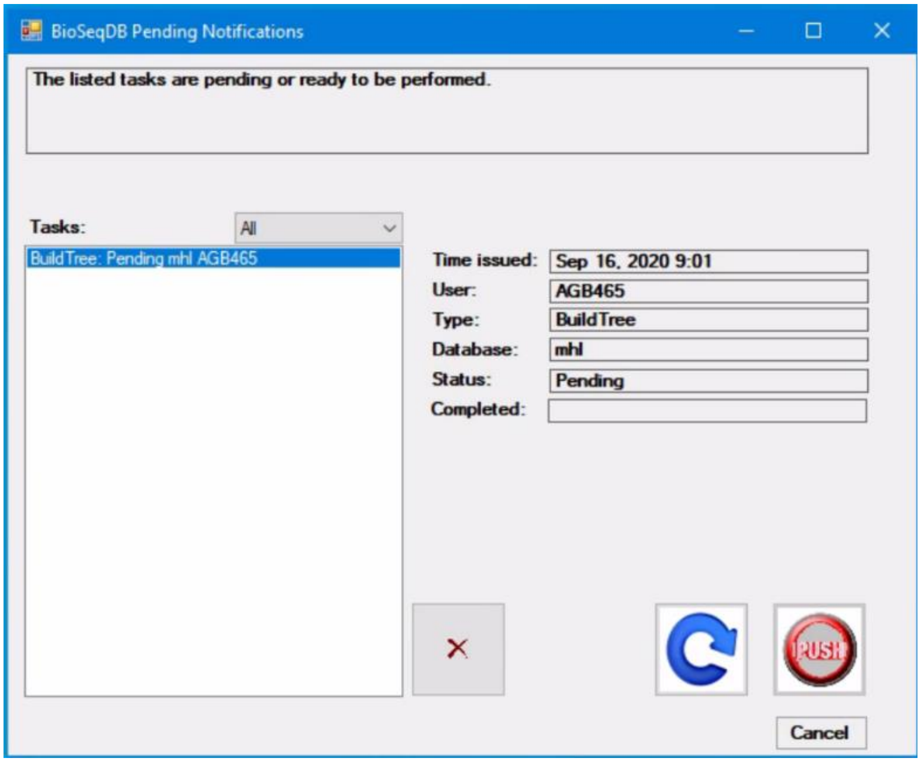
The Gene XRef file describes the target gene name prefixes that are identified in the Abricate results from each of the online virus databases (Card, Ecoli_VF, ECOH, VFDB, NCBI, ResFinder, PlasmidFinder, Argannot). Each line contains a gene name prefix and a description, separated by a comma, as follows:

```
apx, Actinobacillus pleuropneumoniae toxin
apf, Type 4 fimbrae
cpx, Capsular polysaccharide
pap, P fimbrae
sfa, S fimbrae
cnf1, Cytotoxic necrotizing factor 1
hly, Hemolysin
iuc, Aerobactin
iut, Aerobactin
```

Use the Memo field to enter identifying information to facilitate tracking the background task in the Notifications dialog.

Running in the background

The Assemble function and the Build tree function are tasks with the potential to take a long time to complete. Rather than cause BioSeqDB to wait until they complete, they are scheduled to run in the background while other BioSeqDB functions can be performed. The tasks running in the background have a status of ‘Pending’. When they complete, their status changes to ‘Ready’. When a task becomes ‘Ready’, the user is notified through the ‘Pending Notifications’ dialog.



BioSeqDB Pending Notifications

The listed tasks are pending or ready to be performed.

Tasks: All

BuildTree: Pending mhl AGB465

Time issued: Sep 16, 2020 9:01
User: AGB465
Type: BuildTree
Database: mhl
Status: Pending
Completed:

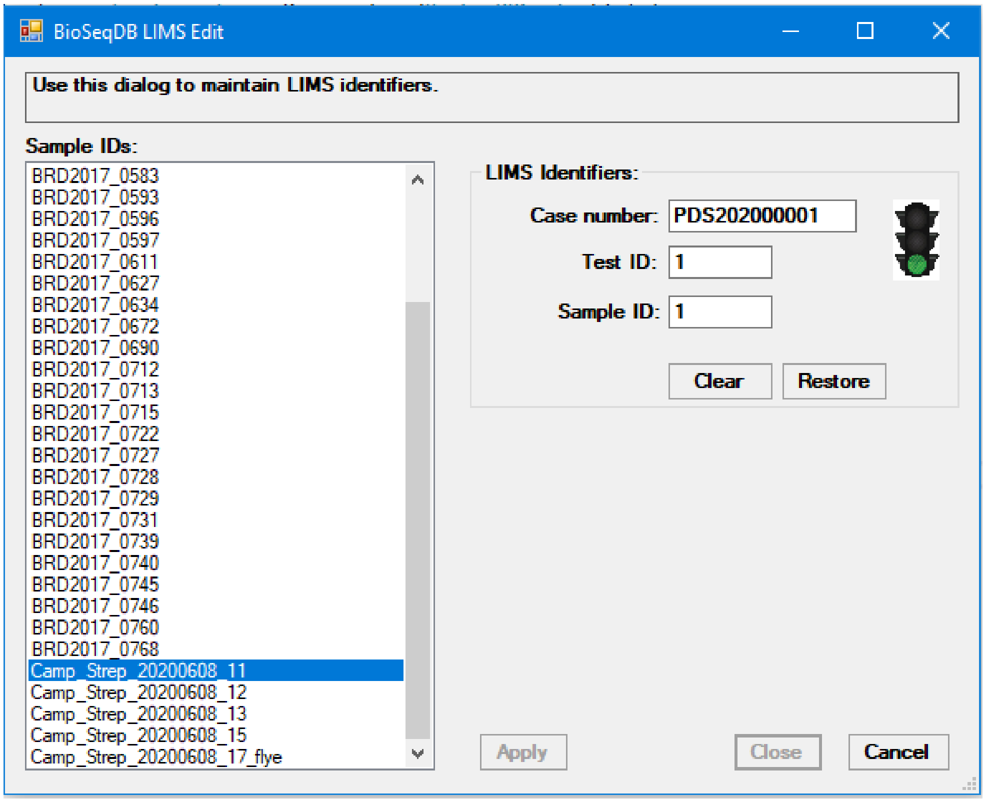
The details of the selected task in the Tasks list are displayed on the right of the dialog. When a task completes, the Status changes to ‘Ready’ and the background of the Status field changes to orange. The time of completion is also reported.

A task, whether the status is ‘Pending’ or ‘Ready’, may be deleted from the list at any time. You can also refresh the list. When a task status changes to ‘Ready’, the ‘Push’ button becomes enabled. By clicking on the ‘Push’ button, the result of the task is displayed, and the task is removed from the task list. For the Build tree function, if successful, the result of the function is displayed as a phylogenetic tree in Dendroscope. For the Assemble function, the results of the assembly are stored in the E:\data\staging\<sample> folder.

The main dialog has a ‘Pending Notifications’ command button that opens the Pending Notifications dialog when clicked. The text of the command button also displays the number of Pending and Ready tasks in the task list of the Pending Notifications dialog.

Editing LIMS identifiers

Any of the existing sequence samples in the currently selected sequence database may be assigned to the LIMS samples via the BioSeqDB LIMS edit dialog:



Use this dialog to maintain LIMS identifiers.

Sample IDs:

BRD2017_0583
BRD2017_0593
BRD2017_0596
BRD2017_0597
BRD2017_0611
BRD2017_0627
BRD2017_0634
BRD2017_0672
BRD2017_0690
BRD2017_0712
BRD2017_0713
BRD2017_0715
BRD2017_0722
BRD2017_0727
BRD2017_0728
BRD2017_0729
BRD2017_0731
BRD2017_0739
BRD2017_0740
BRD2017_0745
BRD2017_0746
BRD2017_0760
BRD2017_0768
Camp_Strep_20200608_11
Camp_Strep_20200608_12
Camp_Strep_20200608_13
Camp_Strep_20200608_15
Camp_Strep_20200608_17_flye

LIMS Identifiers:

Case number: PDS202000001
Test ID: 1
Sample ID: 1

Select a sequence sample from the list on the left, and use the edit are on the right to add, modify or clear any associated values. The traffic light icon indicates when value combinations represent valid values. When 'Apply' is enabled, this indicates that there are potential changes in values that you are editing. When you click on 'Apply', those changes are recorded permanently. The 'Apply' button is only enabled when the LIMS identifier values are valid.

For BioSeqDB administrators

- Control file appsettings.json

  BioSeqDB has its own repository of variables and values that it uses to keep track of user preferences and selections, currently selected databases and outstanding tasks. The file name is appsettings.json and it resides in the same folder as the BioSeqDB service executable on the server (C:\BioSeqDB\Service\appsettings.json). The data structure is mapped by the BioSeqDBConfig class in the BioSeqDBConfig.cs source file in BioSeqDB.

  The basic structure of the appsettings.json file is:

```
{
        <global settings>
        “seqDBs”:
        {
                <list of DBs and their properties>
        }
        “Users”:
        {
                <list of users and their properties>
        }
        “Tasks”:
        {
                <list of tasks and their properties (empty if no outstanding tasks)>
        }
}
```

  Normally you should not make any changes to the values in the appsettings.json file. However, sometimes for diagnostic purposes it is useful to examine the values recorded in this file.

  If updating the executable files for BioSeqDB service, take care not to overwrite the appsettings.json file or the appsettings files for any of the users.

- Updating the executable

  Normally when updating the service executable, only the following files are needed:

  BioSeqService.exe
  BioSeqService.pdb
  BioSeqDB.ModelClient.*
  BioSeqDBData.*

These files are found in the BioSeqDBSolution\BioSeqService\bin\Debug folder of the development environment.

- User ID management

  Currently BioSeqDB uses a quite simple user management system. There is a list of usernames and passwords in the appsettings.json file that represents valid users of BioSeqDB. These names are maintained manually. This is one exception where the appsettings.json file must be edited manually. Simply make whatever changes are necessary to identify the BioSeqDB users and save the changes. The passwords must be manually encrypted for the initial setup, but the user may change their own passwords once they are registered. There is an encryption utility available in the development sandbox project.

  Take care to conform to the syntax of Json to avoid errors at startup.

- Source control

  The BioSeqDB source is stored on GitHub at https://github.com/ArnieBerg/BioSeqDBSolution. The source for the Linux seqdb scripts is stored at https://github.com/jimmyliu1326/seqdb.

FAQ

1. **If I have a standard reference genome defined for the currently selected database, and I do a backup, is the reference genome backed up as well?**

   No, at this point the reference genome is not backed up.

2. **The 'BioSeqDB LIMS edit' dialog allowed me to enter a case number/test ID/sample ID that does not exist in the LIMS. Is that correct?**

   Yes, at this point there is no attempt to cross-reference the values you enter with actual LIMS identifiers.

3. **Which molecular linkage method should I choose for Build Tree?**

   This really depends on how specific you want to define molecular linkage. If you are looking to identify epidemiological linked sequences, you may want to use a strict distance cutoff such that you have 100% confidence that the sequences you identify are very closely related to the queries. However, you must keep in mind that there might be sequences which happened to miss the distance threshold by a small margin and could potentially be highly relevant.

   If you are unaware of a suitable distance threshold to define close relatedness for your query genomes, then using \`tophits\` or \`cluster\` methods are recommended. However, you should also keep in mind that with the \`tophits\` method, because it is a rank-based method, the identified sequences could in fact be highly distant from your queries or there could be a high abundance of closely related sequences, but some were excluded for not meeting the top-ranking threshold.

   Hence, to avoid the exclusion of relevant strains due to user defined thresholds, we introduced the cluster method that attempts to optimize the parameters that should be used to define subpopulations closely related to the queries. However, with the \`cluster\` method, the approach relies on the existence of an underlying population structure within the database. Consequently, this approach is not ideal for small sized databases with limited genetic diversity.

4. **One of my pending tasks stays pending even though I know it completed. How is this possible?**

   This can happen if BioSeqDB is restarted while tasks are pending. It loses track of which tasks are pending and ready. The tasks continue to appear in the notification list, but the status may not be correct. Best practice is to leave BioSeqDB running continuously if possible.

5. **What if the output of one of the background tasks produces a large amount of information?**

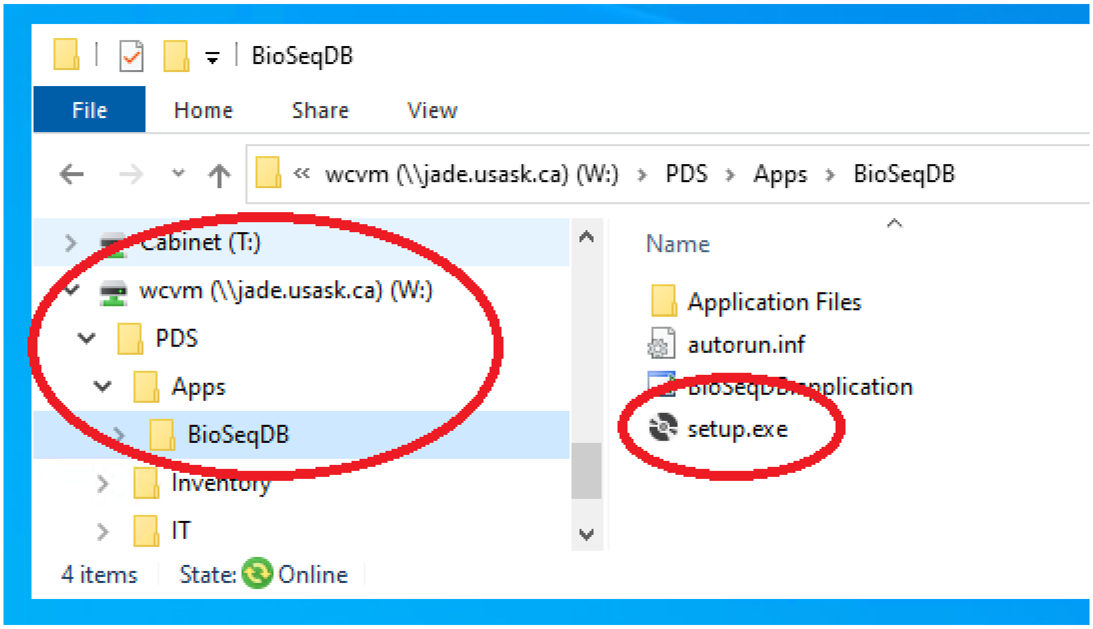
   Normally when a task completes, there is a small amount of information reported as to the progress, success and/or error associated with the task. If the amount of information exceeds the capacity of the dialog, it is stored in a TaskLog file and opened automatically in the default Windows editor.

Appendix A: Deploying the BioSeqDB client

Before deploying the BioSeqDB client to your computer, ensure the following have been done:

1. The BioSeqDB administrator must register your NSID in BioSeqDB.
2. Make sure C:\Temp exists on your computer.
3. Install Dendroscope and set PathToDendroscope in appsettings for user. If Dendroscope is not installed, the output of the Build Tree function cannot be visualized.
4. Make sure Excel is installed. If Excel is not installed, some output results may not be viewable in Excel format.

The BioSeqDB client deployment is handled by an installation method known as ClickOnce. After installing on your computer for the first time, you will have an entry for BioSeqDB in the Start menu. On subsequent times when you launch BioSeqDB, it will automatically check whether a newer version is available and prompt you to install it.



To access the setup file for BioSeqDB, you should have a mapped network drive to W:\\PDS\Apps\BioSeqDB.

Double-click on the **setup.exe** file in the BIoSeqDB folder and click on the prompts to start the installation process. Once the installation is complete, the application will start.

Once the application is installed, all you need to do to keep it up to date is to launch it from the start menu. If a newer version is available, you will be prompted to install it.

You may also want to pin the BioSeqDB app to the taskbar. Do this by right-clicking on the BioSeqDB icon on the taskbar while the app is running and selecting 'Pin to taskbar'.

If for some reason you need to uninstall the app, this may be done from ‘Programs and Features’ in the Control Panel.