BIONUMERICS Tutorial:

Hash-based wgMLST typing

1 Introduction

Hash-based allele calling is a viable option to perform wgMLST without the need for a centrally maintained nomenclature and could therefore still be used after the Applied Maths Cloud Calculation Engine will be taken offline on December 31st, 2024, as part of the BIONUMERICS phase-out (see https://www.bionumerics.com/news/bionumerics-phasing-out).

In the hash-based allele calling algorithm implemented in the *WGS tools local plugin*, allele sequences that fulfill the nomenclature acceptance criteria are converted by a hash function into 17-digit integer numbers. Since hash functions are deterministic, a given allele sequence will always result into the same hash value. However, hash values as such cannot be used directly in the software, because the character experiment type in BIONUMERICS is not designed to reliably store and compare 17-digit integers. Hash values also look very different from the simple integer allele IDs that wgMLST users are familiar with.

To circumvent these issues, the *WGS tools local plugin* sets up a local allele nomenclature in which the hashes are converted into simple integer IDs. The latter are stored in the **wgMLST_Local** experiment type, which is automatically created and synchronized with the **wgMLST** experiment type. Each time a wgMLST assembly-based allele calling job result is retrieved, the **wgMLST_Local** experiment is automatically updated with the hash-based allele calls using the local nomenclature.

The **wgMLST_Local** experiment type only considers assembly-based wgMLST allele calls, assembly-free allele calling is not taken into account.

To allow exchange of wgMLST data between different labs and/or databases, local wgMLST profiles can be exported as hash values, and imported from hash values (see 6).

The process for setting up and running hash-based wgMLST allele calling will be illustrated in this tutorial using the *Listeria monocytogenes* demonstration database.

2 Preparing the database

2.1 Option 1: Download demo database from the Startup Screen

1. Click the  button, located in the toolbar in the *BIONUMERICS Startup* window.

This calls the *Tutorial databases* window (see Figure 1).

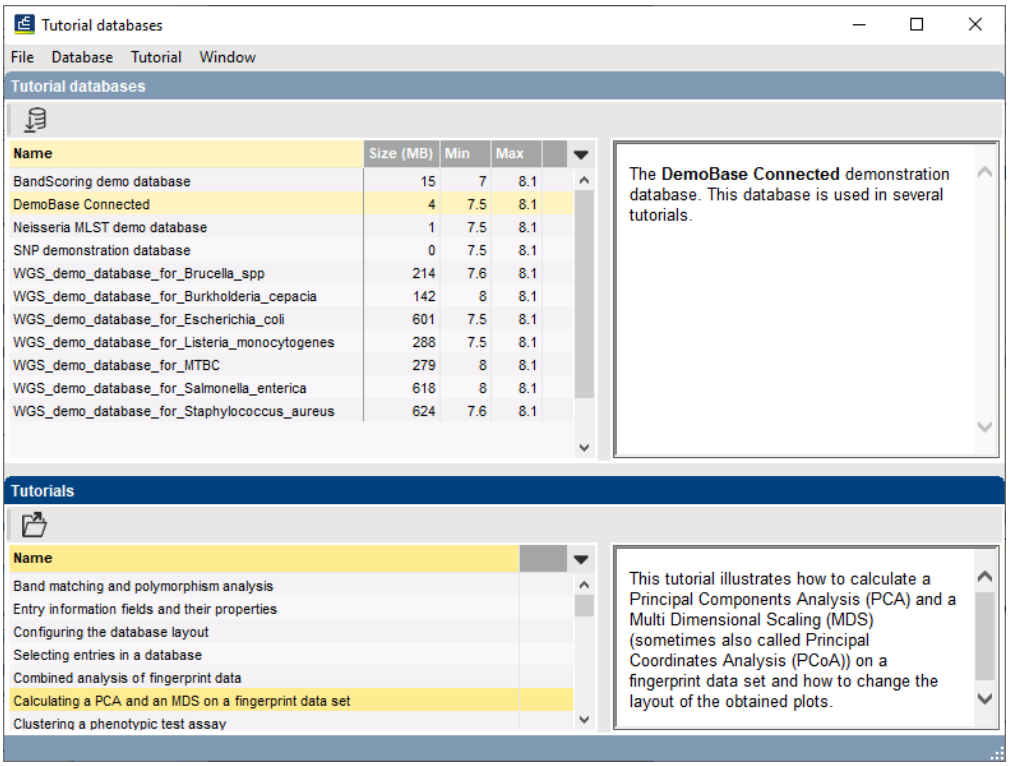

Figure 1: The *Tutorial databases* window, used to download the demonstration database.

2. Select the **WGS_demo_database_for_Listeria_monocytogenes** from the list and select ***Database*** > ***Download*** ( ).

3. Confirm the installation of the database and press <***OK***> after successful installation of the database.

4. Close the *Tutorial databases* window with ***File*** > ***Exit***.

The **WGS_demo_database_for_Listeria_monocytogenes** appears in the *BIONUMERICS Startup* window.

5. Double-click the **WGS_demo_database_for_Listeria_monocytogenes** in the *BIONUMERICS Startup* window to open the database.

2.2 Option 2: Restore demo database from back-up file

A BIONUMERICS back-up file of the WGS demo database for *Listeria monocytogenes* is also available on our website. This backup can be restored to a functional database in BIONUMERICS.

6. Download the file `WGS_LMO1.bnbk` file from https://www.bionumerics.com/download/sample-data, under 'WGS_demo_database_for_Listeria_monocytogenes'.

In contrast to other browsers, some versions of Internet Explorer rename the `WGS_LMO1.bnbk` database backup file into `WGS_LMO1.zip`. If this happens, you should manually remove the `.zip` file extension and replace with `.bnbk`. A warning will appear ("If you change a file name extension, the file might become unusable."), but you can safely confirm this action. Keep in mind that Windows might not display the `.zip` file extension if the option "Hide extensions for known file types" is checked in your Windows folder options.

7. In the *BIONUMERICS Startup* window, press the  button. From the menu that appears, select ***Restore database...***.

8. Browse for the downloaded file and select ***Create copy***. Note that, if ***Overwrite*** is selected, an existing database will be overwritten.

9. Specify a new name for this demonstration database, e.g. “WGS Listeria demobase”.

10. Click <***OK***> to start restoring the database from the backup file (see Figure 2).

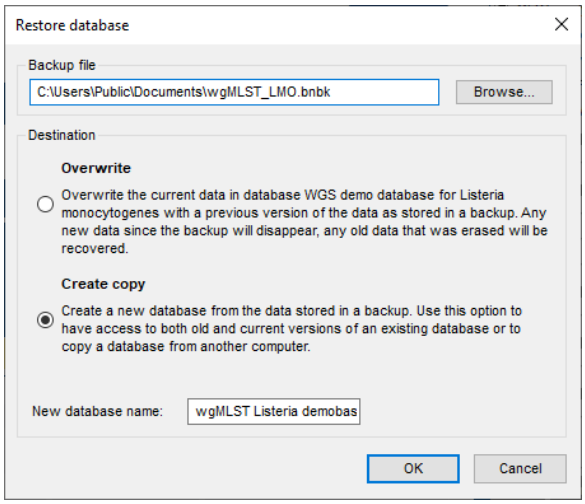

Figure 2: Restoring the WGS demonstration database from the backup file `WGS_LMO1.bnbk`.

11. Once the process is complete, click <***Yes***> to open the database.

The *Main* window is displayed (see Figure 3).

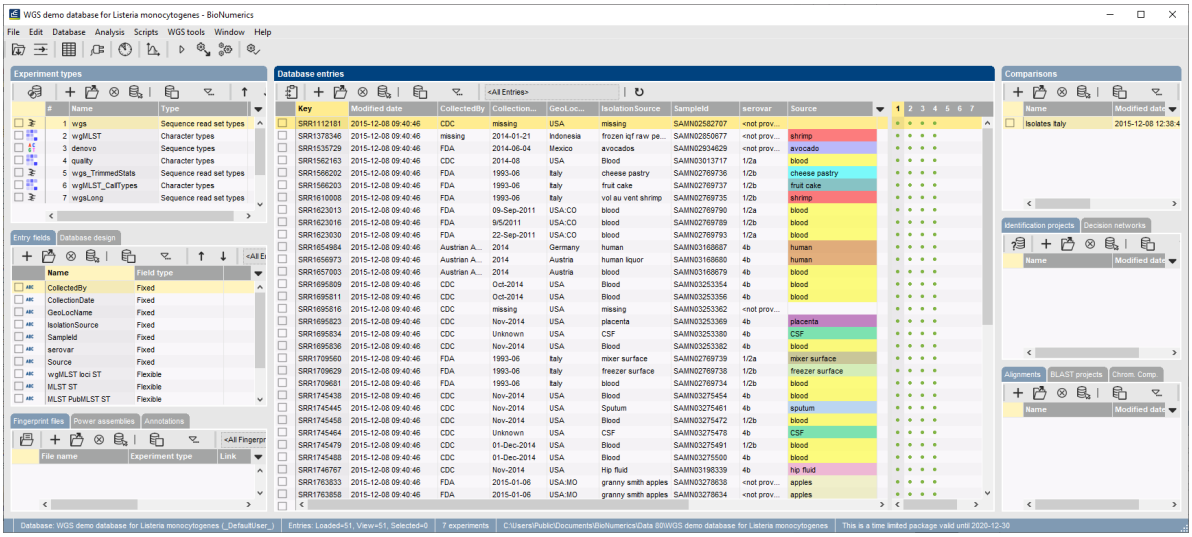

Figure 3: The *Listeria monocytogenes* demonstration database: the *Main* window.

3 About the demonstration database

The WGS Listeria demo database contains links to sequence read set data on NCBI's sequence read archive (SRA) for 51 publicly available sequencing runs. Sequence read set experiment type **wgs** contains the link to the sequence read set data on NCBI (SRA) with some raw data statistics.

The full wgMLST analysis (de novo assembly, assembly-based calls and assembly-free calls) was performed on this set of samples using default settings and the *L. monocytogenes* wgMLST scheme on the Applied Maths Calculation Engine.

1. Select ***WGS tools*** > ***Settings...*** to access the settings of the plugin.

The calculation engine project is linked to the *Listeria monocytogenes* allele database. No credits are assigned to this project so no jobs can be submitted to the external calculation engine, however since the option ***Enable running jobs on my own computer*** is checked in the *Calculation engine* tab, it is possible to run jobs on your own computer (see Figure 4).

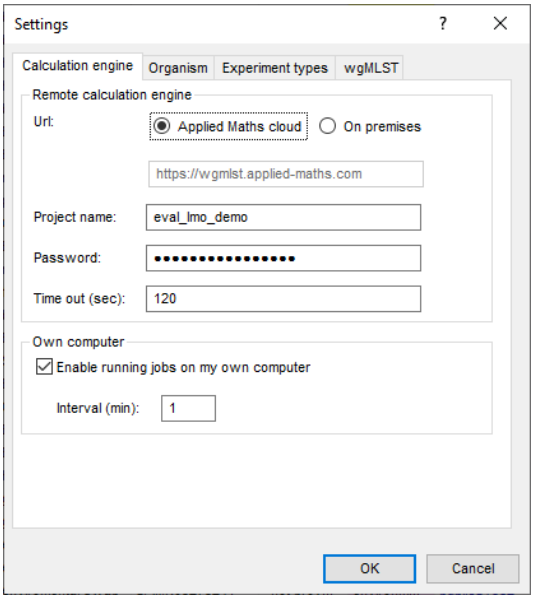

Figure 4: The *Calculation engine* tab of the *Calculation engine settings* dialog box.

2. Click on the *wgMLST tab* (see Figure 5) and press the <***Auto submission criteria***> button (see Figure 6).

By default, the ***Use nomenclature acceptance criteria*** option will be checked, meaning that the automatic submission settings are defined by the curator of the allele database.

3. Click <***Cancel***> twice to close the *Calculation engine settings* dialog box.

Experiment types linked to wgMLST analysis are present in the database for each of the entries and are displayed in the *Experiment types* panel (see Figure 7):

- Character experiment type **wgMLST** contains the allele calls for detected loci in each sample, where the consensus from assembly-based and assembly-free calling resulted in a single allele ID.
- Sequence experiment type **denovo** contains the results from the de novo assembly algorithm, i.e. concatenated de novo contig sequences.

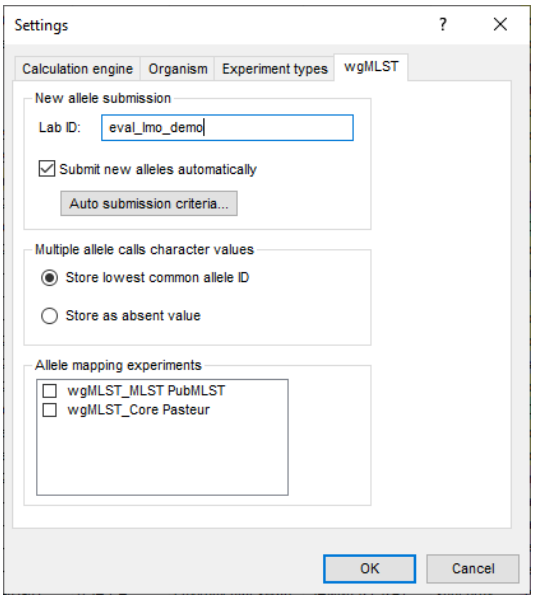

Figure 5: The *wgMLST tab* of the *Calculation engine settings* dialog box.

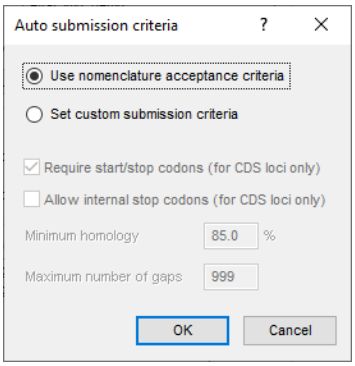

Figure 6: The *Auto submission criteria* dialog box.

- Character experiment type **quality** contains quality statistics for the raw data, the de novo assembly and the different allele identification algorithms.
- Sequence read set experiment type **wgs_TrimmedStats**: contains some data statistics about the reads retained after trimming.
- Character experiment type **wgMLST_CallTypes**: contains details on the call types.

No data is available for the sequence read set type **wgsLong** in the demo database. This sequence read set is used to store links to long read sequence read data (e.g. PacBio or MinION datasets).

Additional information, stored in entry info fields (CollectionDate, CollectedBy, serovar, etc.) was collected from the corresponding publications and added to the demonstration database.

By clicking on one of the green dots next to an entry in the database, the corresponding results can be viewed, either in a separate window or in an experiment card for the character data types:

4. Click on the green colored dot for one of the entries in the first column in the *Experiment presence* panel. Column 1 corresponds to the first experiment type listed in the *Experiment types* panel, which is **wgs** in the default configuration.

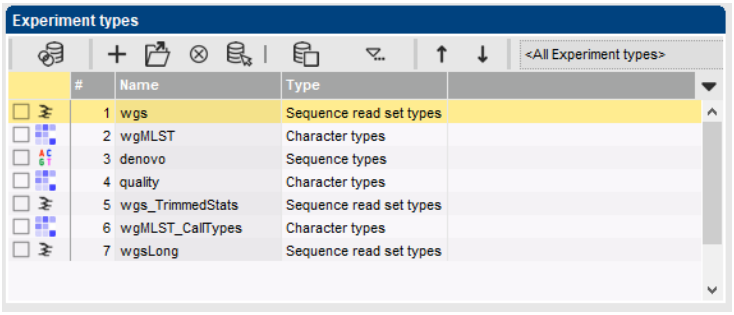

Figure 7: The *Experiment types* panel of the *Main* window.

In the *Sequence read set experiment* window, the link to the sequence read set data on NCBI (SRA) with a summary of the characteristics of the sequence read set is displayed: *Read set size*, *Sequence length statistics*, *Quality statistics*, *Base statistics* (see Figure 8).

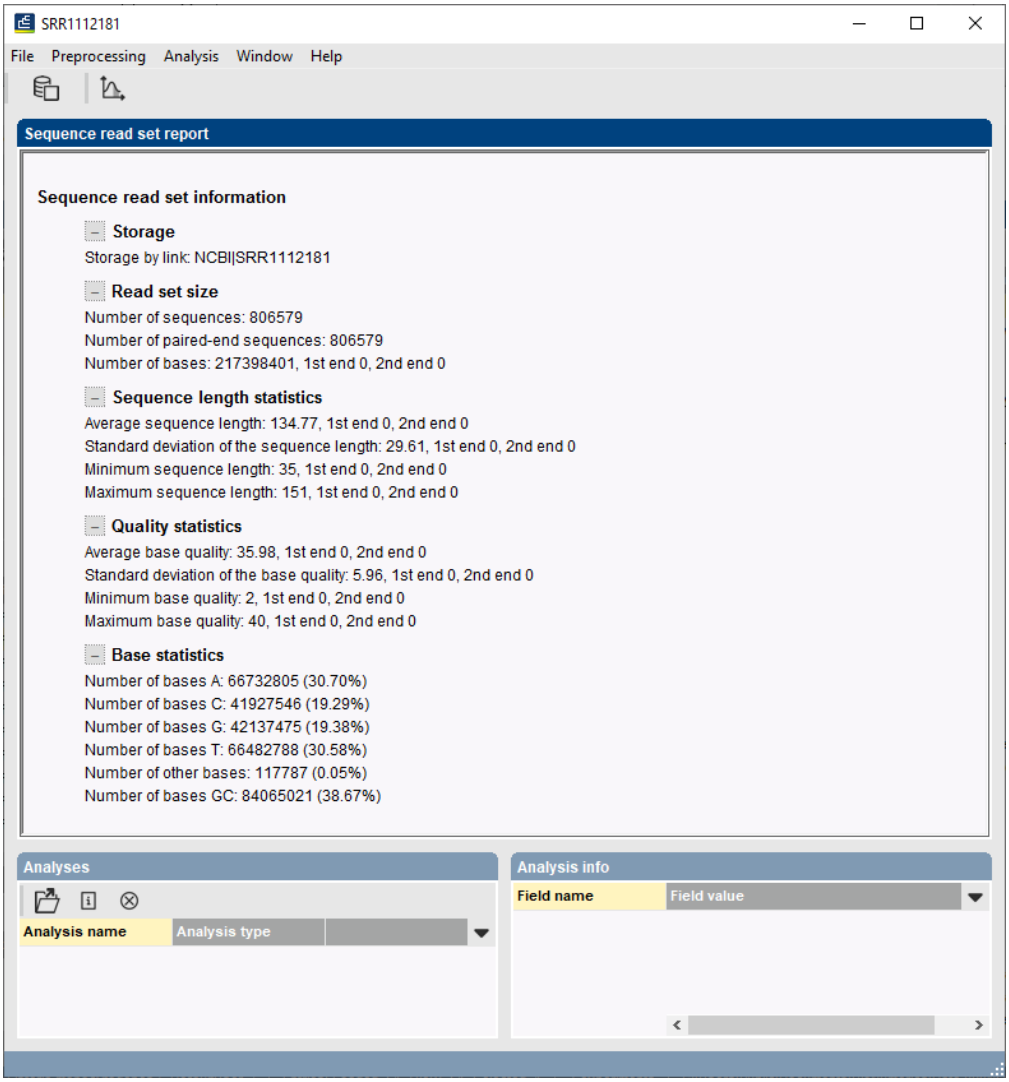

Figure 8: The sequence read set experiment card for an entry.

5. Close the *Sequence read set experiment* window.

6. Click on the green colored dot for one of the entries in the second column in the *Experiment presence* panel. Column 2 corresponds to the second experiment type listed in the *Experiment types* panel, which is **wgMLST** in the default configuration.

Character experiment type **wgMLST** contains the allele calls for detected loci in each sample, where the consensus from assembly-based and assembly-free calling resulted in a single allele ID (see Figure 9).

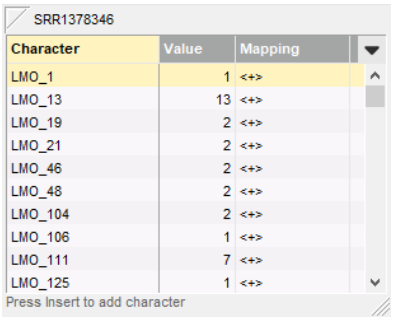

Figure 9: The character experiment card for an entry.

7. Close the character experiment card by clicking on the triangle in the top left corner.

8. Click on the green colored dot for one of the entries in the third column in the *Experiment presence* panel. Column 3 corresponds to the third experiment type listed in the *Experiment types* panel, which is **denovo** in the default configuration.

The *Sequence editor* window opens, containing the results from the de novo assembly algorithm, i.e. concatenated de novo contig sequences (see Figure 10).

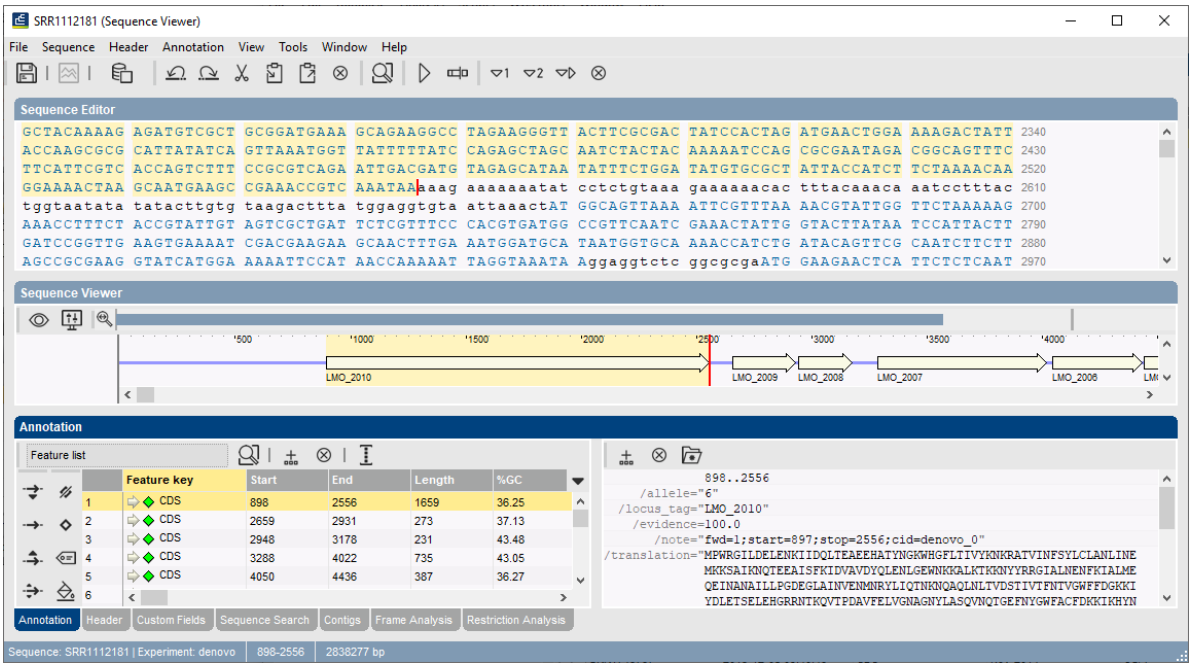

Figure 10: The *Sequence editor* window.

9. Close the *Sequence editor* window.

10. Click on the green colored dot in column 4 to open the **quality** character card (default configuration) for an entry in the database.

The **quality** character card contains quality statistics for the raw data, the de novo assembly and the different allele identification algorithms (see Figure 11).

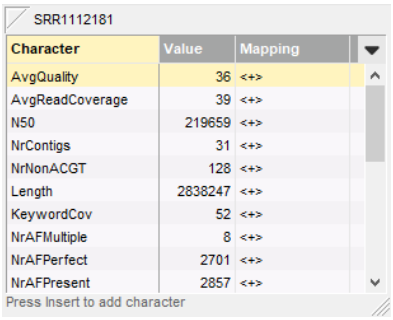

Figure 11: The character experiment card for an entry.

11. Close the character experiment card by clicking on the triangle in the top left corner.

4 Installation of the WGS tools local plugin

The *WGS tools local plugin* is designed to work with the latest BIONUMERICS version. Earlier versions are not tested an might have compatibility issues. Please check our software download page (https://www.bionumerics.com/download/software) to see if an update to your current version is available.

The *WGS tools local plugin* should be installed in a BIONUMERICS database in which the *WGS tools plugin* is already installed and connected to a Calculation Engine instance via a CE project and password.

Installation of the *WGS tools local plugin* should be done before December 31th, 2024 while the Applied Maths Cloud Calculation Engine is still up and running.

The *WGS tools local plugin* is made available as an online plugin, which can be installed in the relational database. This has as an advantage that no Windows administrator rights are required for installation. Furthermore, in a multi-user database setup, this procedure ensures that all database users work with the same plugin version.

Proceed as follows to install the *WGS tools local plugin*:

1. Select ***File*** > ***Install / remove plugins...*** () in the *Main* window to call the *Plugins and Scripts* dialog box.

2. Press the <***Manage database plugins***> button to open the *Manage database plugins* dialog box.

The *Manage database plugins* dialog box lists the plugins that are currently stored in the relational database. Likely, this list is initially empty.

3. Press the <***Add/Update***> button to open the *Add database plugins* dialog box (see Figure 12.

4. Check the check box in front of the *WGS tools local plugin* and click <***OK***>.

A message appears, indicating that the plugin will be loaded after the database is restarted.

5. Press <***Close***> in the *Add database plugins* dialog box and repeat the same action in the *Plugins and Scripts* dialog box.

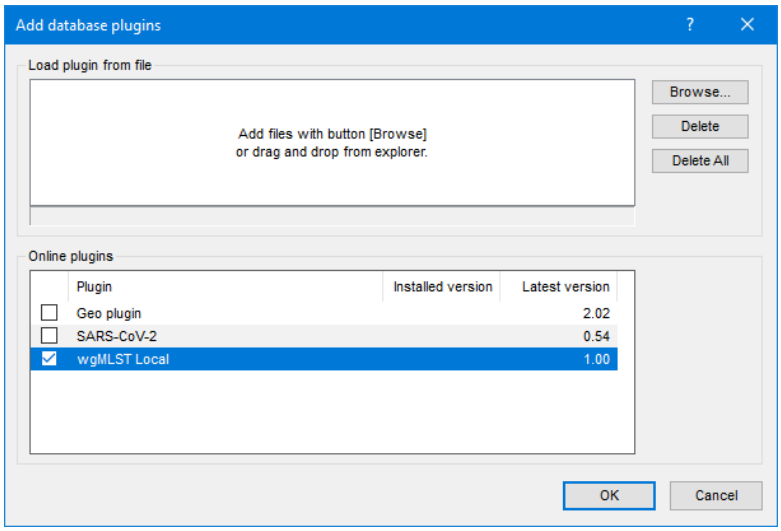

Figure 12: The *Add database plugins* dialog box, listing the *WGS tools local plugin*.

6. Close and restart the BIONUMERICS database.

7. Select ***File*** > ***Install / remove plugins...*** ( ) in the *Main* window to call the *Plugins and Scripts* dialog box again.

The *WGS tools local plugin* is now displayed in the *Plugins tab* of the *Plugins and Scripts* dialog box at the bottom of the list and is preceded by a database icon (see Figure 13).

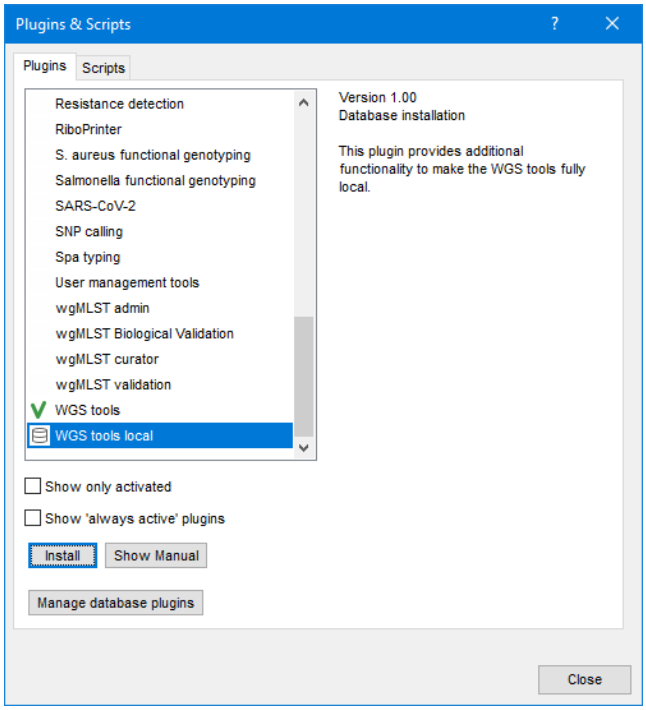

Figure 13: The *Plugins and Scripts* dialog box with the *WGS tools local plugin* highlighted.

8. Click on the *WGS tools local plugin* to highlight it, press <***Install***> and confirm the plugin installation.

The plugin automatically creates and initiates an experiment type **wgMLST_Local** by performing

a synchronization with the Calculation Engine.

The *WGS tools local plugin* will use the experiment type that is specified to contain the wgMLST allele calls in the *Experiment types panel* of the *Calculation engine settings* dialog box and will add the **_Local** suffix to the experiment name. Even though the actual name might be different, the experiment type will be referred to as **wgMLST_Local** for reasons of conciseness.

9. When the synchronization is complete, press <***OK***> to close the message box.

A message box pops up with the question ”Do you want to initialize the local allele nomenclature with the accepted alleles?”.

Answering “No” will create a local wgMLST nomenclature from scratch: allele IDs are assigned starting from 1 for each locus.

Answering “Yes” will start the local nomenclature from the current accepted alleles in the central nomenclature and new alleles will be added by increasing integer identifiers.

When the assembly-based accepted alleles are next updated, there might be newly accepted alleles with IDs that have meanwhile already been assigned to (other) local alleles. Inevitably, from the point of initialization on, the allele IDs will start to diverge!

If the assembly-based wgMLST search data are not up-to-date, an update of the search data will be performed first. This may take several minutes.

10. Select <***Yes***>.

This action will create a file `wgMLST_Local_Nomenclature.txt` in the source files directory with the local nomenclature.

A notification appears when the process is completed, indicating that the plugin is installed and prompting to restart the database.

11. Press <***Close***> in the *Plugins and Scripts* dialog box and close and restart the BIONUMERICS database.

5 (Re-)running assembly-based wgMLST calls

From the *Experiment presence* panel in the *Main* window, it can be seen that there are currently no experiments present for **wgMLST_Local**. To populate **wgMLST_Local** experiments for all entries in the database, it is required to re-run all assembly-based wgMLST calls. On the local calculation engine, this is a relatively quick process that does not require CE credits. For extremely large databases, we recommend submitting the jobs in batches of maximum 1000 entries.

1. Select all 51 entries in the *Listeria monocytogenes* demonstration database using ***Edit*** > ***Select all*** (**Ctrl+A**).

2. Select ***WGS tools*** > ***Submit jobs...*** (▷).

3. In the *Submit jobs* dialog box (see Figure 14), make sure ***Own computer*** is checked to run the jobs on the local calculation engine, which does not require CE credits.

4. Check ***wgMLST assembly-based calls*** under ***Algorithms*** and ***Re-submit already processed data*** to make sure that the jobs are re-run.

5. Press the <***OK***> button to submit the jobs.

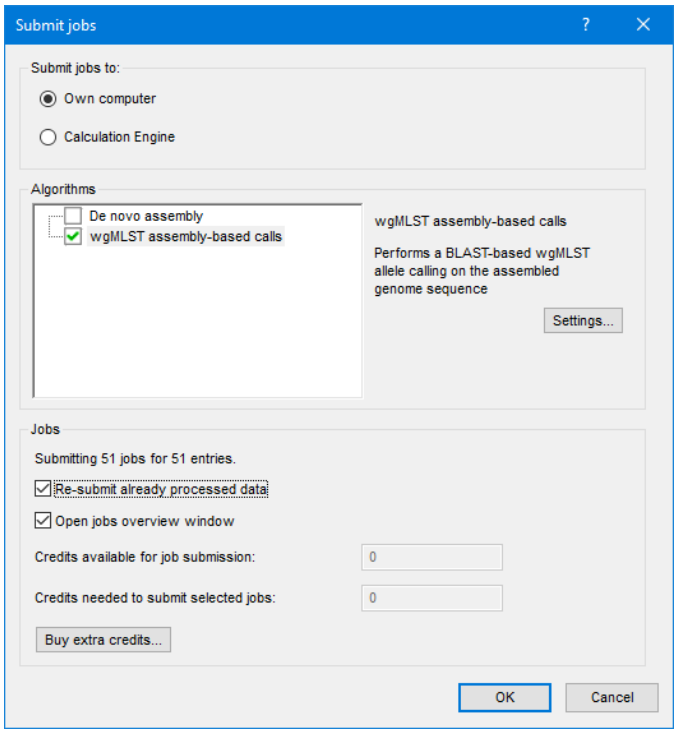

Figure 14: The *Submit jobs* dialog box, ready to re-run all wgMLST assembly-based allele calling jobs on the local calculation engine.

If the local assembly-based wgMLST search data are not up-to-date, they will be updated first. This might take several minutes. Next, the jobs are submitted to the local calculation engine.

6. If not already open, select ***WGS tools*** > ***Jobs overview...*** ( ) to open the *Job overview* window.

The assembly-based allele calling jobs complete relatively fast. In our test setup, the 51 jobs finished in about 25 minutes and it took another 5 minutes to retrieve the job results and store them in the BIONUMERICS database.

7. Select ***View*** > ***Refresh*** (, **F5**) to check the execution progress.

8. When the jobs are completed, highlight them in the *Job overview* window and select ***Jobs*** > ***Get results*** ( ) to retrieve the results.

The *Experiment presence* panel now shows a green dot for the **wgMLST_Local** experiment type with each entry in the database.

9. Click on a green dot for the **wgMLST_Local** experiment of a random entry and on one for the **wgMLST** experiment of the same entry to display their *Experiment card* window (see Figure 15).

Since we started the local nomenclature from the accepted alleles of the central nomenclature (see 4), both wgMLST profiles look almost identical. Small differences can occur (as illustrated in Figure 15 for locus LMO_49) because assembly-free allele calls are taken into account for **wgMLST**, while **wgMLST_Local** uses exclusively assembly-based allele calls. As time progresses, both nomenclatures will start to diverge and more differences will be observed between both experiments because of new alleles that are assigned different IDs. When the Calculation Engine goes offline, new alleles will not be called for **wgMLST** and the profiles will become incomplete. In contrast, the wgMLST profiles stored in **wgMLST_Local** will remain complete because the hash-based allele calling will continue to call new alleles.

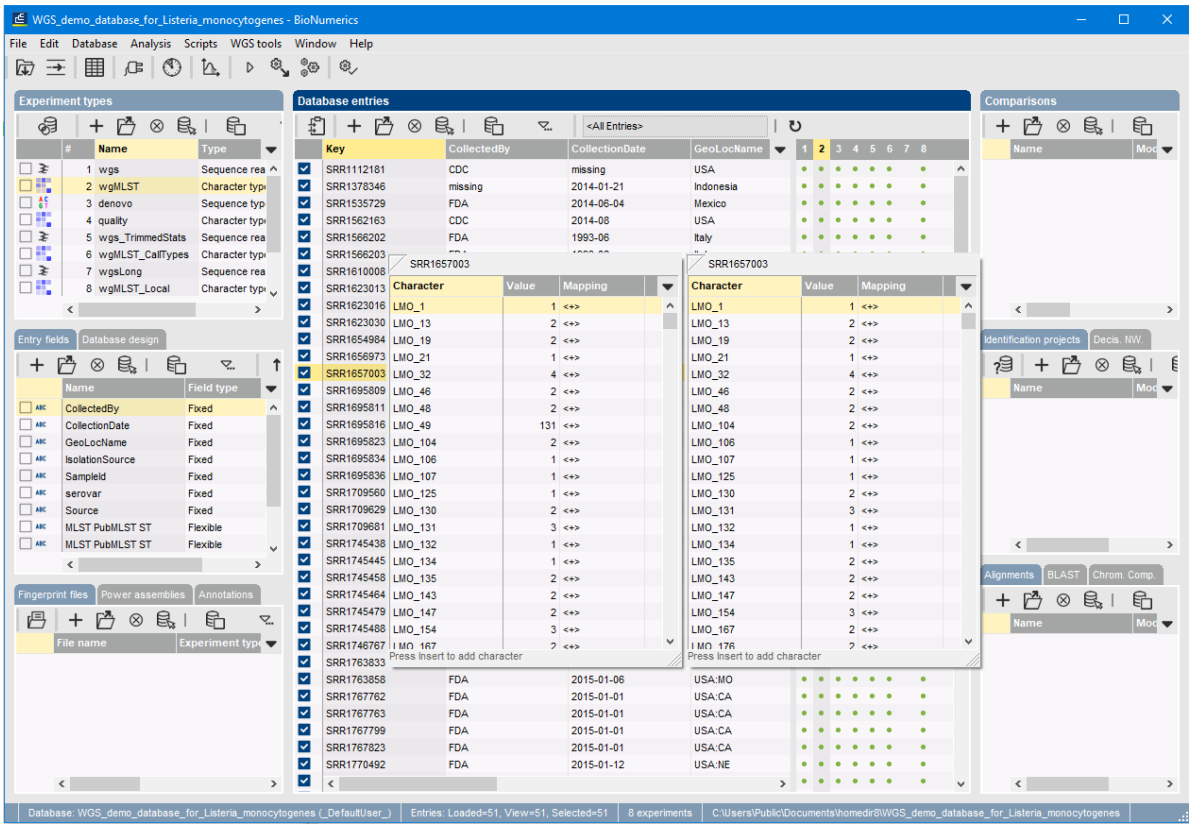

Figure 15: The *Main* window with an *Experiment card* window for **wgMLST** (left) and **wgMLST_Local** (right) for entry SRR1657003. Note that locus LMO_49 is called differently in both profiles.

The **wgMLST_Local** experiments can be analyzed with the same methods and the results interpreted in the same way as **wgMLST** experiments.

6 Import and export of hash-based calls

Because a *local* nomenclature is used, **wgMLST_Local** allelic profiles from different BIONUMERICS databases cannot be directly compared with each other. In order to exchange and compare data with other databases or labs, the allele IDs need to be converted into something that can be uniquely referenced, i.e. the hash values obtained from the allele sequence.

Proceed as follows to export allele hashes for a set of entries:

1. Select the entries that you want to export and use ***File*** > ***Export...***.

2. In the *Export* dialog box, under the topic ***Character type data***, highlight ***Export fields and hashed wgMLST calls*** and press <***Next***>.

3. In the *Export* dialog box that appears, select the ***Fields*** and the subset of the **wgMLST_Local** experiment (under ***Characters***) that you wish to include in the export. Press <***Next***>.

4. In the *Settings* dialog box, specify how absent values are denoted and whether or not the export should be limited to the active characters only. Press <***Finish***>.

This action will create the `export.csv` in the database directory and will open the file in your computer's default editor for `*.csv` files.

MS Excel, which is the default CSV editor on most PCs, displays the allele hashes in scientific notation (e.g. 5,83E+16). For categorical data, this does in fact not make sense.

By specifying “No” for the preference ***Export table files in CSV format***, allele hashes files will be exported in tab-delimited text format (see the Reference manual, Chapter The BIONUMERICS user interface).

Import should be done in another BIONUMERICS database in which the *WGS tools plugin* and the *WGS tools local plugin* are both installed. The installation procedure for the *WGS tools plugin* is covered in the corresponding plugin manual and the *WGS tools local plugin* is described in 4, so the procedures will not be repeated here.

Proceed as follows to import entries with their allele hashes from an exported text file:

5. Select ***File*** > ***Import...*** (, **Ctrl+I**).

6. Browse for the text file (`*.csv` or `*.txt`) exported earlier, highlight the option ***Import fields and hashed wgMLST calls (text file)*** and press <***Finish***>.

7. Press <***Next***>.

Since no import template is available yet, we will need to create one first. We assume in this tutorial that the other database also uses SRA accessions as database key:

8. Double-click on ’Key’ in the *Import rules* dialog box, select ***Key*** as destination in the *Edit data destination* dialog box (see Figure 16) and press <***OK***>.

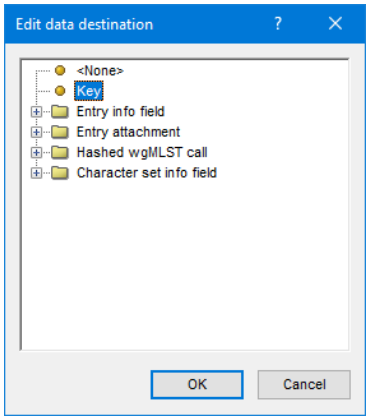

Figure 16: The *Edit data destination* dialog box for the ’Key’ field.

9. Highlight all wgMLST loci in the *Import rules* dialog box and press the <***Edit destination...***> button.

10. In the *Edit data destination* dialog box, select ***wgMLST_Local*** under ***Hashed wgMLST call*** (see Figure 17).

11. Press <***Next***> in the *Import rules* dialog box.

12. Leave ***Key*** checked and press <***Finish***> in the *Import links* dialog box.

13. Enter e.g. “Hashes” as ***Name*** for the import template and press <***OK***>.

Now that an import template is created, we can use it in this and future imports of hashed wgMLST allele calls.

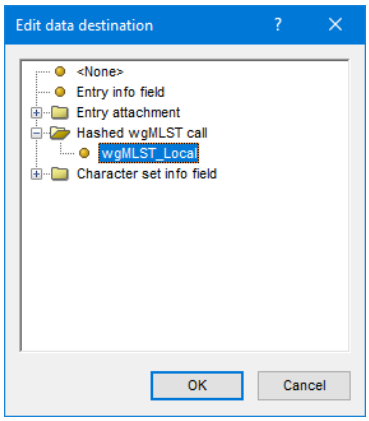

Figure 17: The *Edit data destination* dialog box for all wgMLST loci in the export file.

14. Press <***Next***> in the *Import template* dialog box to apply the **Hashes** import template.

15. Press <***Finish***> to import the hash-based allele calls.