BIOMÉRIEUX

BIONUMERICS®

version 8 - PLUGINS

Custom genotyping plugin

Contents

NOTES

SUPPORT BY APPLIED MATHS, A BIOMÉRIEUX COMPANY

While the best efforts have been made in preparing this manuscript, no liability is assumed by the authors with respect to the use of the information provided.

Applied Maths, a bioMérieux company, will provide support to research laboratories in developing new and highly specialized applications, as well as to diagnostic laboratories where speed, efficiency and continuity are of primary importance. Our software thanks its current status for a part to the response of many customers worldwide. Please contact us if you have any problems or questions concerning the use of BIONUMERICS®, or suggestions for improvement, refinement or extension of the software to your specific applications:

Applied Maths NV
Keistraat 120
9830 Sint-Martens-Latem
Belgium
PHONE: +32 9 2222 100
FAX: +32 9 2222 102
E-MAIL: BE-DAU-INFO@biomerieux.com
URL: https://www.bionumerics.com

Applied Maths, Inc.
11940 Jollyville Road, Suite 115N
Austin, Texas 78759
U.S.A.
PHONE: +1 512-482-9700
FAX: +1 512-482-9708
E-MAIL: US-DAU-INFO@biomerieux.com

LIMITATIONS ON USE

The BIONUMERICS® software, its plugin tools and their accompanying guides are subject to the terms and conditions outlined in the License Agreement. The support, entitlement to upgrades and the right to use the software automatically terminate if the user fails to comply with any of the statements of the License Agreement. No part of this guide may be reproduced by any means without prior written permission of the authors.

BIONUMERICS® uses following third-party software tools and libraries:

- Python 3.8 release from the Python Software Foundation, https://www.python.org/
- Xerces library for XML input and output from the Apache Software Foundation, https://xerces.apache.org/
- NCBI toolkit version 2.11.0, https://www.ncbi.nlm.nih.gov/BLAST/
- SRA Toolkit, https://ncbi.github.io/sra-tools/
- Boost c++ libraries, https://www.boost.org/
- Samtools for interacting with SAM / BAM files, https://www.htslib.org/download/
- 7-Zip (7za.exe), https://www.7-zip.org/
- Zlib library, https://zlib.net/
- Pigz for parallel gzip compression, https://zlib.net/pigz/
- Cairo 2D graphics library version 1.12.14, https://cairographics.org/
- Crypto++ library version 5.5.2, https://www.cryptopp.com/
- OpenSSL library, https://www.openssl.org/
- libSVM library for Support Vector Machines, https://www.csie.ntu.edu.tw/~cjlin/libsvm/
- SQLite version 3.7.17, https://www.sqlite.org/
- pymzML Python module version 2.4.7, https://github.com/pymzml/pymzML
- NumPy Python library version 1.19.1, https://www.numpy.org/
- BioPython Python library version 1.78, https://www.biopython.org/
- pyodbc Python module version 4.0.30, https://pypi.org/project/pyodbc/
- jinja2 Python library version 2.11.2, https://pypi.org/project/Jinja2/
- MarkupSafe Python library version 1.1.1, https://pypi.org/project/MarkupSafe/
- regex Python library version 2.5.91, https://pypi.org/project/regex/
- Chromium Embedded Framework, https://bitbucket.org/chromiumembedded/cef/wiki/Home
- SPAdes genome assembler version 3.15.3, https://bioinf.spbau.ru/spades *
- SKESA version 2.3.0, https://github.com/ncbi/SKESA/releases
- Unicycler version 0.5.0, https://github.com/rrwick/Unicycler/releases *
- Velvet for Windows, source code can be downloaded from https://www.bionumerics.com/download/open-source
- Bowtie2 version 2.2.5 (https://bowtie-bio.sourceforge.net/bowtie2/index.shtml)*
- SNAP version 2.0.0, https://www.microsoft.com/en-us/research/project/snap/
- RAxML version 8.2.11, https://github.com/stamatak/standard-RAxML/releases

- FastTree version 2.1.10, https://www.microbesonline.org/fasttree/
- CFSAN SNP pipeline version 2.2.0, https://github.com/CFSAN-Biostatistics/snp-pipeline *
- Prokka version 1.14.5, https://github.com/tseemann/prokka *
- sourmash version 4.1.0, https://github.com/dib-lab/sourmash **
- SeqSero2 for Windows, source code can be downloaded from https://www.bionumerics.com/download/open-source
- Fastp version 0.22.0, https://github.com/OpenGene/fastp

*: On Calculation Engine only **: See license conditions below

Sourmash license conditions:

Chapter 1

Starting and setting up BIONUMERICS

1.1 Introduction

The *Custom genotyping plugin* allows you to detect and extract sequences from genome sequences using a BLAST or in silico PCR approach. Additionally, it allows the detection of mutations using a BLAST approach and the confirmation of species identity using sourmash [1].

While all organism-specific genotyping plugins and the *Resistance detection plugin* work with precompiled online knowledge bases, the *Custom genotyping plugin* allows you to use your own custom knowledge bases. This means that, with an appropriately defined knowledge base, the software can screen genomes for any set of user-defined sequences or traits e.g. antibiotic resistance or virulence.

The *Custom genotyping plugin* is supported in the **BIONUMERICS-SEQ** and **BIONUMERICS-SUITE** configurations.

1.2 Startup program

Make sure the latest version of BIONUMERICS is installed (https://www.bionumerics.com/download/software). The installation manual can be downloaded from https://www.bionumerics.com/download/manuals.

When BIONUMERICS is launched from the Windows start panel or when the BIONUMERICS shortcut () on your computer's desktop is double-clicked, the **Startup program** is run. This program shows the *BIONUMERICS Startup* window (see Figure 1.1).

A new BIONUMERICS database is created from the Startup program by pressing the  button.

An existing database is opened in BIONUMERICS with  or by simply double-clicking on a database name in the list.

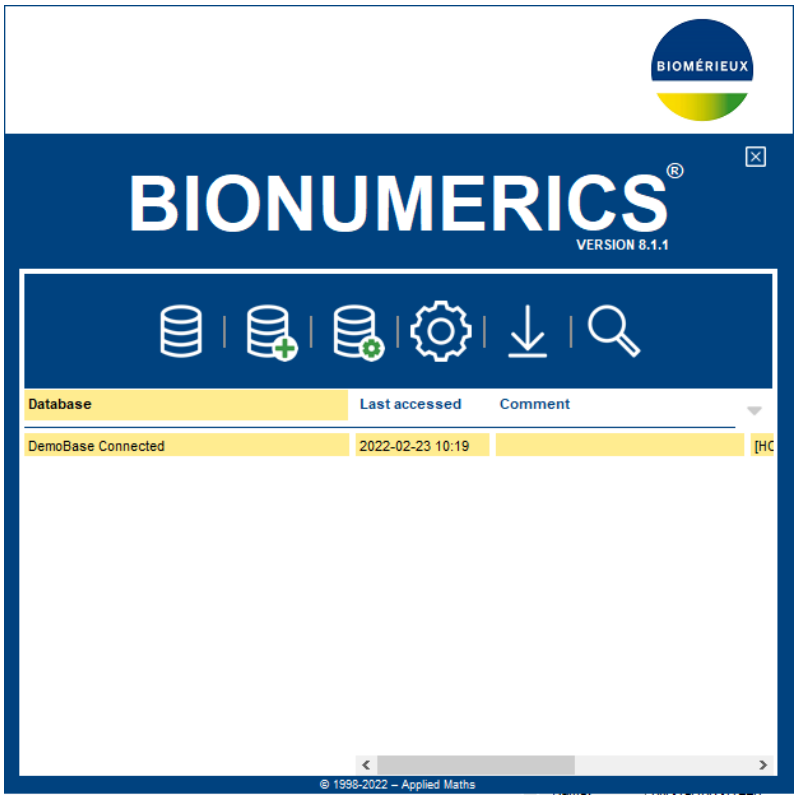

Figure 1.1: The *BIONUMERICS Startup* window.

1.3 Installation of the Custom genotyping plugin

Proceed as follows to install the *Custom genotyping plugin*:

3.1 Call the *Plugins and Scripts* dialog box from the *Main* window with ***File*** > ***Install / remove plugins...*** ( ).

3.2 Select the *Custom genotyping plugin* from the list and press the <***Install***> button.

3.3 Confirm the installation of the plugin.

3.4 Press <***Yes***> to confirm that you want to change the plugin settings.

A message appears, confirming the installation of the plugin and prompting you to restart BIONUMERICS.

3.5 Press <***OK***> in the confirmation message.

3.6 Press <***Close***> to close the *Plugins and Scripts* dialog box.

3.7 Close and re-open the database to complete the installation of the plugin.

The *Custom genotyping plugin* installs menu items in the main menu of the software under ***Genotyping*** (see Figure 1.2).

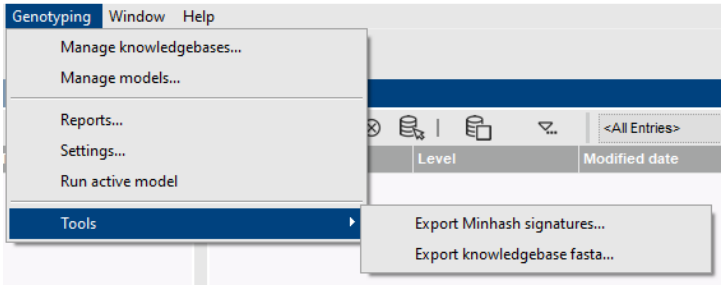

Figure 1.2: New menu items, available after installation of the *Custom genotyping plugin*.

Chapter 2

Genotyping knowledge bases

2.1 Introduction

All custom genotyping plugin features make use of a knowledge base of some kind. Knowledge bases are at the heart of functional genotyping because they literally contain the knowledge on how to interpret genome sequences in function of the feature they were designed for.

The fact that custom knowledge bases can be created and used is the main advantage of the *Custom genotyping plugin*. Depending on the type of knowledge base (e.g. BLAST-based or PCR-based) and depending on the genotyping feature, the knowledge bases need to be in a specific format (see 2.2). Example knowledge bases can easily be created by the plugin to facilitate the creation of your own custom knowledge base (see 2.3).

2.2 Knowledge base format

2.2.1 Required and optional files

A BLAST-based knowledge base, i.e. applicable for sequence detection, sequence extraction and acquired trait detection features contains following files:

- An info file (see 2.2.2)
- A sequences file (see 2.2.3)
- An optional trimming patterns file (see 2.2.4)
- An optional mutational loci file (see 2.2.5)
- An optional mutations file (see 2.2.6)

A PCR-based knowledge base, i.e. applicable for in silico PCR detection and in silico PCR extraction features contains following files:

- An info file (see 2.2.2)
- A primers file (see 2.2.7)

A Minhash-based knowledge base, i.e. applicable for identification using sourmash contains following files:

- An info file (see 2.2.2)
- A sourmash parameters file (see 2.2.8)
- A sourmash signatures file (see 2.2.9)
- A genomes info file (see 2.2.10)

2.2.2 Info file

The `info.json` file is a JSON format (https://en.wikipedia.org/wiki/JSON) file that must, at minimum, contain string values for the keys **name** and **version**. It may optionally contain a key **description** with a string value. It may also optionally contain a key **changelog**, which is an object mapping strings to strings. Each key in the changelog object corresponds to a version, and each value a description of what happened for that version.

Example:

```
{
"version": "2.0",
"name": "BLAST based example knowledgebase",
"description": "An example knowledgebase for BLAST based features",
"changelog": {
"1.0": "First version",
"1.1": "Fixed typo in locus name, added some traits",
"2.0": "Added several loci sequences, and more alleles of existing loci"
}
}
```

2.2.3 Sequences file

The `sequences.fasta` file is a FASTA format (https://en.wikipedia.org/wiki/FASTA_format) file containing nucleotide sequences. The sequences may be alleles of loci, plasmids, phages, partial CDS, etc.. The header of each sequence (line starting with $>$) uses a key-value pair convention specific for BIONUMERICS genotyping knowledge bases. Each pair is separated by a pipe character (|), and keys are separated from values by a single equals sign ($=$).

Non-tagged, pipe separated fields in the `sequences.fasta` are ignored.

The following keys are possible:

- **identifier:** Always required. Must be unique within the file!
- **name:** Always required. Multiple sequences may use the same name, allowing you to provide multiple alleles of the same locus for instance. For the sequence extraction feature, the best matching allele is used as starting point.
- **trait:** Only required for the trait detection feature. Multiple key-value pairs may be given for trait, allowing you to link a single sequence with multiple traits.

- **accession:** Optional. The accession corresponding to the sequence. GenBank and RefSeq accessions may be automatically hyperlinked in the report.
- **publication:** Optional. The publication source of the sequence. PubMed, PubMed Central, and DOI identifiers may be automatically hyperlinked in the report. If this field is present in the knowledge base, the field can be displayed in the ***Complete*** template of the report.
- **description:** Optional. If this field is present in the knowledge base, the field can be displayed in the ***Complete*** template of the report.

Example:

```
>identifier=unique_id_1|name=name_of_sequence_or_locus|@trait=trait1
ACGT
>identifier=unique_id_2|name=name_of_sequence_or_locus|@trait=trait1|@trait=trait2|@trait=trait3
ACGT
```

To provide multiple values for trait, it must be marked as an array with the '@' character. You can then repeat the tag multiple times to add more values. The '@' character will be stripped from the name when used internally. A file must consistently use either the array or singular form of the trait tag. Values will be stored internally in a list, even if there is only one value.

2.2.4 Trimming patterns file

The `trimming_patterns.tsv` file is a TSV format (https://en.wikipedia.org/wiki/Tab-separated_values) file that describes patterns used in the trimming of sequences. The file has a fixed set of required columns. Values in some columns are optional. If no values are given in those columns, default values are used.

```
-----==FWD==>-------<==REV==-----
```

The table must have the following columns:

- **identifier:** The unique identifier, which should match the identifier of a sequence or primer pair definition depending on its use.
- **start_pattern:** The 5' pattern on the sense strand relative to the reference sequence. May contain ambiguous (IUPAC) codes.
- **stop_pattern:** The 3' pattern on the sense strand relative to the reference sequence. May contain ambiguous (IUPAC) codes. Note how this differs from the reverse primer of PCR, which is defined for the 5' of the anti-sense strand relative to the reference sequence.
- **reference_length:** the length of the trimmed reference sequence, including the trimming pattern matching ends.
- **max_length_offset:** the maximum allowed size difference between a trimming corrected sequence length and the trimmed reference length. Empty values are interpreted as no maximum. Note that empty is different from zero; zero means no length difference is allowed.
- **start_offset:** the offset from the start trimming pattern outermost position. Positive numbers trim downstream, negative numbers extend upstream. The 'auto' keyword can be used to trim the whole pattern, and combined with +/-X to further trim or extend from there. Similar to starting from the innermost position.

- **stop_offset:** The offset from the stop trimming pattern outermost position. Positive numbers extend downstream, negative numbers trim upstream. The 'auto' keyword can be used to trim the whole pattern, and combined with +/-X to further trim or extend from there. Similar to starting from the innermost position.
- **level:** the type of sequence on which the trimming acts, converting between them if needed. Currently only the 'DNA' level is supported.

Patterns with IUPAC codes are converted to regular expressions. The regular expressions match the corresponding bases as well as other valid IUPAC codes. For example, `N` would match `A`, `C`, `T` or `G`, but also `Y` (the ambiguous IUPAC code for `CT`) and more.

Example:

```
identifier start_pattern stop_pattern reference_length max_length_offset start_offset stop_offset
level unique_id_1 ACGT ACGT 12 0 2 auto-2 DNA unique_id_2 ACGT ACGT 12 0 0 0 DNA
```

2.2.5 Mutational loci file

The `mutational_loci.fasta` file is a FASTA (https://en.wikipedia.org/wiki/FASTA_format) format file containing nucleotide sequences. The sequences may be any reference sequence, but for loci where mutations are sought on a protein level, those sequences should be valid ORFs. The header of each sequence (i.e., the line starting with $>$) uses a key-value pair convention specific for BIONUMERICS genotyping knowledgebases. Each pair is separated by a pipe character ("|"), and keys are separated from values by a single equals sign ("="). The following keys are possible:

- **identifier**: Always required and must be unique within the file.
- **locus**: Always required and must be unique within the file.

Example:

```
>identifier=mut_ref_unique_id_1|locus=EX1
GATTAAGTTATTAAGGGCGCACGGTGGATGCCTTGGCACTAGAAGCC
```

2.2.6 Mutations file

The `mutations.tsv` file is a is a Tab Separated Values (TSV) formatted file (https://en.wikipedia.org/wiki/Tab-separated_values) containing known mutations about loci in the mutational loci file (2.2.5). This file is optional, it is only required if you intend to detect known mutations and/or their associated traits. The following columns *must* be present:

- **identifier**: The unique identifier, in the specific BIONUMERICS mutational identifier format (see below).
- **required**: A logical expression (optional, see below). A logical combination of related mutations that must be fulfilled for the trait associated with this mutation to be present.
- **trait**: The trait resulting from the presence of the mutation and, if applicable, the 'required' expression evaluating to True.

- **publication**: Optional. The publication source of the sequence. PubMed, PubMed Central and DOI identifiers may be automatically hyperlinked in the report. If given, may be displayed in the extra rows of tables displayed in 'Complete' reports.
- **description**: Optional. If given, may be displayed in the extra rows of tables displayed in 'Complete' reports.

Example:

```
identifier required trait publication description
EX1_dT3A Example resistance Literature search, Jan-April 2022
EX1_dC32CG Example resistance Literature search, Jan-April 2022
EX1_dG19- Example susceptibility A public database, 2021-02-01
EX1_dA43T (EX1_dT3A & EX1_dC32CG) | (!  EX1_dG19- ) Example stronger resistance Self determined
using BIONUMERICS
```

Mutational identifiers

Identifiers unambiguously describe a mutation by combining required information in the form '{locus}_{level}{ref}{pos}{mut}'.

Fields:

- 'locus' accepts numbers, upper and lowercase letters and underscores.
- 'level' accepts 'p' or 'd' for respectively protein and DNA level mutations. It defaults to 'p'. A level for RNA ('r') is not currently supported.
- 'ref' (reference) only accepts one uppercase letter or asterisk ('*').
- 'pos' (position) only accepts numbers. Uses the one-based coordinate system.
- 'mut' (mutation) accepts one or more uppercase letters, a dash ('-') or asterisk ('*').

Remarks:

- Multiple letters in a mutation indicates insertions.
- An asterisk ('*') is used for stop codons, and is only valid on protein level.
- Mutation can be a dash ('-') for a deletion.
- Insertions should start at the last non-mutated reference position.
- Mutation and reference may be identical! This can be useful to require a conservation of a position in logical expressions.
- Identifiers do not need to be listed in the identifier column of the mutations file to be used in expressions.

Example: 'lacA_dT65TAG': an insertion of nucleotides A and G after position 65, which has nucleotide T, in locus lacA.

Logical expressions

Expressions combine variables identifiers with operands to describe a logical boolean expression whose result depends on the values associated with the identifiers.

Operands:

- Equals sign ('=') for 'EQUALS'
- Pipe ('|') for 'OR'
- Ampersand ('&') for 'AND'
- Exclamation mark ('!') for 'NOT'
- Caret ('∧') for 'XOR'
- Brackets ('(' and ')') for grouping

Remarks:

- A single identifier with no operands also counts as an expression, and evaluates to the value of the identifier.
- Expressions are interpreted as normal infix expressions. But there is no operator precedence which may lead to confusing results if insufficient brackets are used.

Example: '((A & B & C) | (X & Y)) & (! N)': A, B and C must be True and/or X and Y must be True, and additionally N may never be True.

Undecided values

Aside from 'True' and 'False', a result may also be 'Undecided'. This result moves through the decision tree in a special way. It may be irrelevant compared to other inputs and stop propagating, or be decisive for the result and propagate further. See the table below for examples.

Contributing variables

The final evaluation of an expression leads to a top-level result, that may contain any combination of variables and lower-level results. We may want to find out which variables contributed to the end result.

In order to find those variables, we descend the tree recursively. We check for each input that led to a result, whether flipping its value changes the result. If so, that input contributes. If none of the values appear to contribute individually, we can assume that all inputs contribute equally. Once we encounter a contributing variable (leaf in the tree), we save the name of that variable. We continue until we have all variable names.

We ignore 'Undecided' values entirely. As long as the end result of the tree is anything other than 'Undecided', an input of that value can never have been decisive for its result and thus cannot have contributed to it.

For example:

2.2.7 Primers file

The `primers.tsv` file is a TSV format (https://en.wikipedia.org/wiki/Tab-separated_values) file describing primers used for in silico PCR. When defining the primer sequences, keep in mind that the reverse primer must match the opposite strand relative to the forward primer, like so:

```
5'--==FWD==>-----------------3'
3'-----------------<==REV==---5'
```

The table must have the following columns:

Operation	Values	Result	Contributing
A AND B	True, False	False	B
A AND B	True, True	True	A and B
A AND B	False, False	False	A and B
A AND B	True, Undecided	Undecided	A
A OR B	True, False	True	A
A OR B	True, True	True	A and B
A OR B	False, False	False	A and B
A OR B	True, Undecided	True	A
NOT A	True	False	A
NOT A	False	True	A
NOT A	Undecided	Undecided	A

- **identifier:** unique identifier of the primer pair definition
- **primer_fwd:** sequence of the forward primer, based on the sense strand of the reference
- **primer_rev:** sequence of the reverse primer, based on the anti-sense strand of the reference
- **reference_length:** length of the expected/reference amplicon, including the primers themselves
- **max_length_offset:** maximum allowed size difference between an amplicon length and the reference amplicon length. Empty values are interpreted as no maximum. Note that empty is different from zero; zero means no length difference is allowed.
- **max_iupac:** maximum ambiguous (IUPAC) bases allowed in the alignment of each primer with the query (default 0)
- **max_mismatch:** maximum number of mismatches allowed in the alignment of each primer with the query (default 0)

Example:

```
identifier primer_fwd primer_rev reference_length max_length_offset max_iupac max_mismatch primer_pair_
ACGTACGT TGCATGCA 100 200 0 0 primer_pair_2 ACGTACGT TGCATGCA 200 200 0 0
```

2.2.8 Sourmash parameters file

The `sourmash_params.json` file is a JSON format (https://en.wikipedia.org/wiki/JSON) file that includes the sourmash kmer size and scaling factor.

Example:

```
{
"kmer_size":  31,
"scaling_factor":  5000
}
```

2.2.9 Sourmash signature file

The `genomes.sig` file is a JSON format (https://en.wikipedia.org/wiki/JSON) file that includes the minhash signatures. It can be generated with the sourmash sketch function or via ***Genotyping***

> ***Tools*** > ***Export Minhash signatures...*** (see 7.2). The latter command will label each signature with the filename `entry_key.fasta`. Additionally, you need to copy the used kmer size and scaling factor into the sourmash parameters file (see 2.2.8).

2.2.10 Genomes info file

The `genome_info.tsv` file is a Tab Separated Values (TSV) formatted file (https://en.wikipedia.org/wiki/Tab-separated_values) that includes the following columns:

- **identifier**: Always required and must be unique within the file. Must be equal to the signature file name (see 2.2.9 without the `.fasta` extension. This corresponds to the entry key from the BIONUMERICS database when the signatures are created with ***Genotyping*** > ***Tools*** > ***Export Minhash signatures...***.
- **genome_accession**: Optional. The accession number corresponding to the sequence. GenBank and RefSeq accessions may be automatically hyperlinked in the report.
- **genome_name**: description of the genome
- **taxid**: taxid number (https://www.ncbi.nlm.nih.gov/taxonomy)
- **genus_name**: genus name
- **genus_threshold**: a mash containment threshold between 0.0 and 100.0. However, thresholds lower than 80 are not recommended as the accuracy of minhashing-based similarity becomes unreliable at this level.
- **species_name**: species name (”-” if unknown)
- **species_threshold**: a mash containment threshold between 0.0 and 100.0 or ”-” if species is unknown. Should be smaller or equal than the genus threshold.
- **subspecies_name**: subspecies name (”-” if unknown)
- **subspecies_threshold**: a mash containment threshold between 0.0 and 100.0 or ”-” if subspecies is unknown. Should be smaller or equal than the species threshold.

Example:

```
CP033092 CP033092.2 Escherichia coli DSM 30083 = JCM 1649 = ATCC 11775 866789 Escherichia 93
coli 95 - -
CP026802 CP026802.1 Shigella sonnei strain ATCC 29930 624 Shigella 99 sonnei 99 - -
CP025979 CP025979.1 Escherichia marmotae strain HT073016 1499973 Escherichia 93 marmotae 95
fake_subspecies 98
```

2.3 Managing knowledge bases

Before a genotyping model can be created, at least one knowledge base should be available.

Select ***Genotyping*** > ***Manage knowledgebases...*** to open the *Manage knowledge bases* dialog box (see Figure 2.1).

The *Manage knowledge bases* dialog box shows all currently available genotyping knowledge bases in the BIONUMERICS database. Following information is displayed about the knowledge bases:

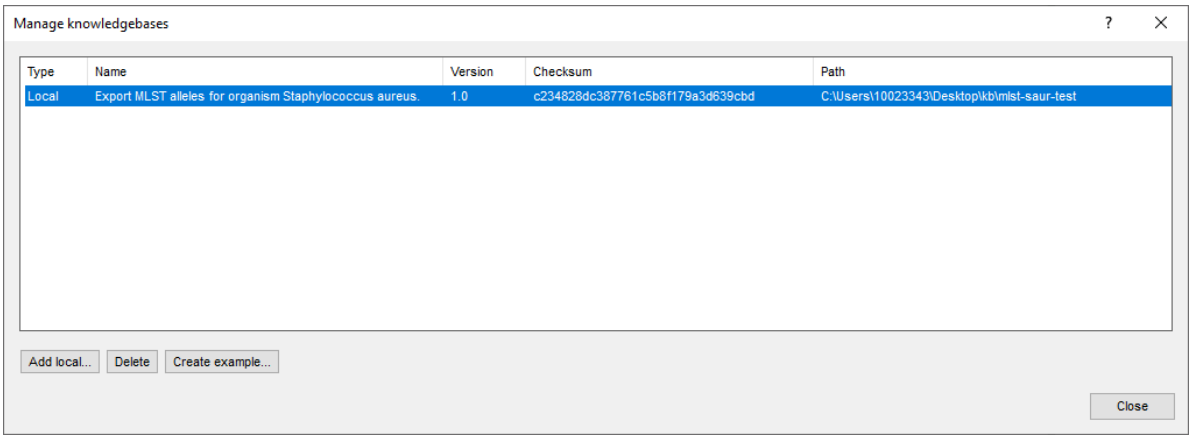

Figure 2.1: The *Manage knowledge bases* dialog box, showing a single knowledge base.

- 'Type': either Online or Local. The knowledge base type will always be Local for the *Custom genotyping plugin*.
- 'Name': the name of the knowledge base, as obtained from the info file? (see 2.2.2).
- 'Version': the knowledge base version, as obtained from the info file (see 2.2.2).
- 'Checksum': MD5 checksum, used to verify the integrity of the local knowledge base.
- 'Path': path to the knowledge base directory.

Initially, this dialog is empty.

Pressing <***Add local...***> displays the *Register local knowledge base* dialog box (see Figure 2.2).

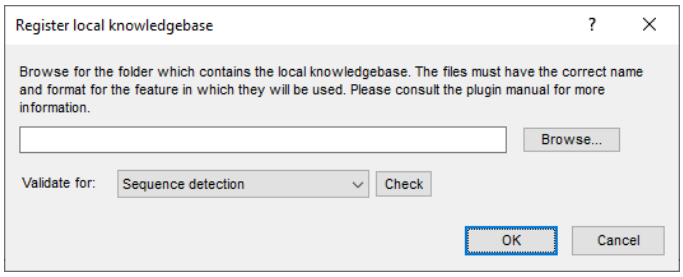

Figure 2.2: The *Register local knowledge base* dialog box.

Press <***Browse...***> and browse for a directory containing all required knowledge base files (see 2.2 for a description of the knowledge base format).

Optionally, the knowledge base can be validated before it is added. Select the type of feature for which the knowledge base is intended for (see 3.1) and press <***Check***>.

Any detected issues are reported in the *Knowledge base validation issues* dialog box (see Figure 2.3).

An error means that the knowledge base is not usable for the selected feature type: a different knowledge base should be specified or the knowledge base's files should be corrected according to the error description. It will not be possible to add a knowledge base with a validation error.

A knowledge base for which only warning messages are raised might be usable for the selected feature, but not all options of the feature are applicable.

Figure 2.3: The *Knowledge base validation issues* dialog box, showing an error and a warning message.

Pressing <***OK***> in the *Register local knowledge base* dialog box will add the validated knowledge base to the list in the *Manage knowledge bases* dialog box.

Optionally, for illustration of the required knowledge base format (see 2.2), an example knowledge base can be generated with <***Create example...***>. This opens the *Create example knowledge base* dialog box (see Figure 2.4).

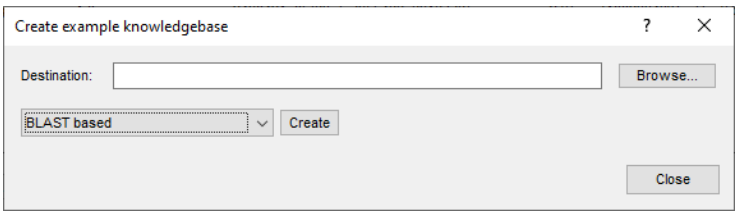

Figure 2.4: The *Create example knowledge base* dialog box.

Three example knowledge bases are available: ***BLAST based***, ***PCR based*** and ***minhash based***. With <***Browse...***>, a directory can be specified in which the knowledge base is created. Press <***Create***> to create the selected example knowledge base. The software automatically opens the knowledge base in Windows Explorer. A `README.md` file in the example knowledge base further explains the format.

Press <***Close***> to close the *Create example knowledge base* dialog box.

To close the *Manage knowledge bases* dialog box, press <***Close***>.

Chapter 3

Managing genotyping models

3.1 Introduction

A genotyping model defines which genotyping analyses (i.e. features) will be executed as well as which knowledge bases and other feature settings will be used during the execution of the model. A user can create several models and switch between these models to change the active model. Only one model can be active, and hence be executed, at the same time.

A feature is one of the following types:

- **Sequence detection**: Uses BLAST to detect target sequences within an input genome sequence. The results are presented in a report.
- **Sequence extraction**: Similar as sequence detection, but additionally the matched sequences are extracted from the genome sequence and stored in their own experiment types.
- **Acquired traits detection**: Similar as sequence detection, but the sequences are linked to phenotypic traits. These traits are reported.
- **Mutation scanning**: Uses BLAST to detect mutations in an input genome sequence. The mutations are presented in a report.
- **Mutational traits detection**: Similar as mutation scanning, but the mutations are linked to phenotypic traits. These traits are reported.
- **In-Silico PCR detection**: Uses an in silico PCR approach to detect target sequences within an input genome sequence.
- **In-Silico PCR extraction**: Similar as in silico PCR detection, but additionally the matched sequences are extracted from the genome sequence and stored in their own experiment types.
- **Species Confirmation**: Uses sourmash [1] to confirm genus, species and (if available) subspecies identity against reference genome signatures in a knowledge base.

Each feature has its settings, which are saved with the model. Probably the most important setting of any feature is the knowledge base it uses.

3.2 Creating a genotyping model

Select ***Genotyping*** > ***Manage models...***. This action will show the *Manage genotyping models* dialog box (see Figure 3.1).

Prior to creating a genotyping model, make sure to add a knowledge base for each of the features.

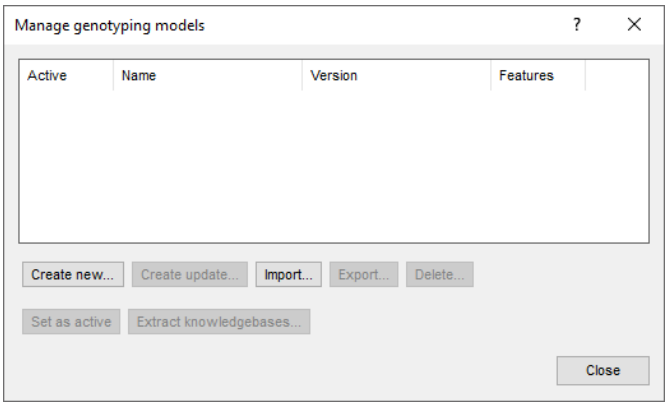

Figure 3.1: The *Manage genotyping models* dialog box.

The *Manage genotyping models* dialog box lists all genotyping models available in the BIONUMERICS database, with their 'Name', 'Version' and the number of features present in the model ('Features'). Only one of the available models is the *active model*, i.e. the model that can be executed. Initially, this list shows up empty.

Press <***Create new...***> to start the *Create model* wizard (see Figure 3.2).

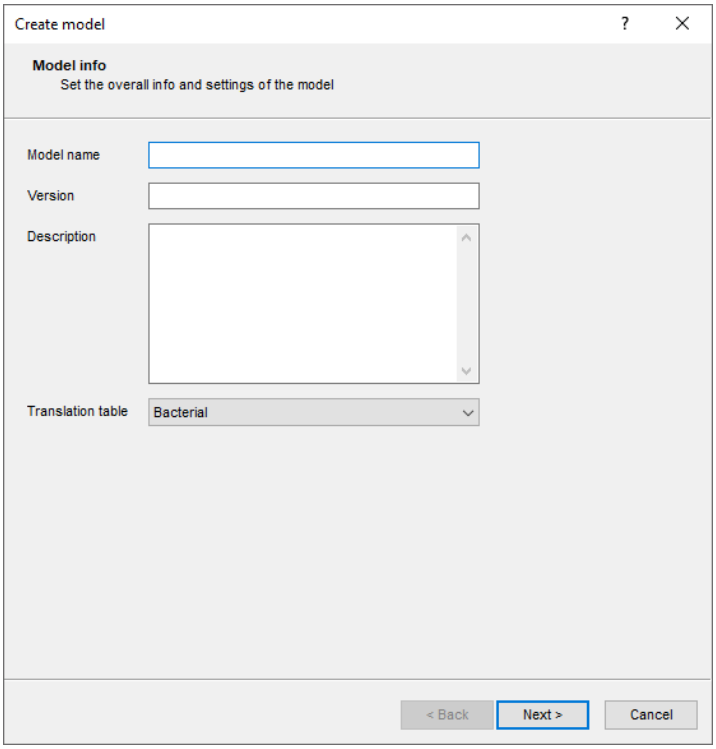

Figure 3.2: The *Create model* wizard, page 1.

A ***Model name*** and ***Version*** are required and this combination should be unique for each model. Optionally, a ***Description*** can be entered for the model.

A ***Translation table*** should be specified, which defaults to 'Bacterial'.

Press <***Next***> to proceed to the second page of the *Create model* wizard.

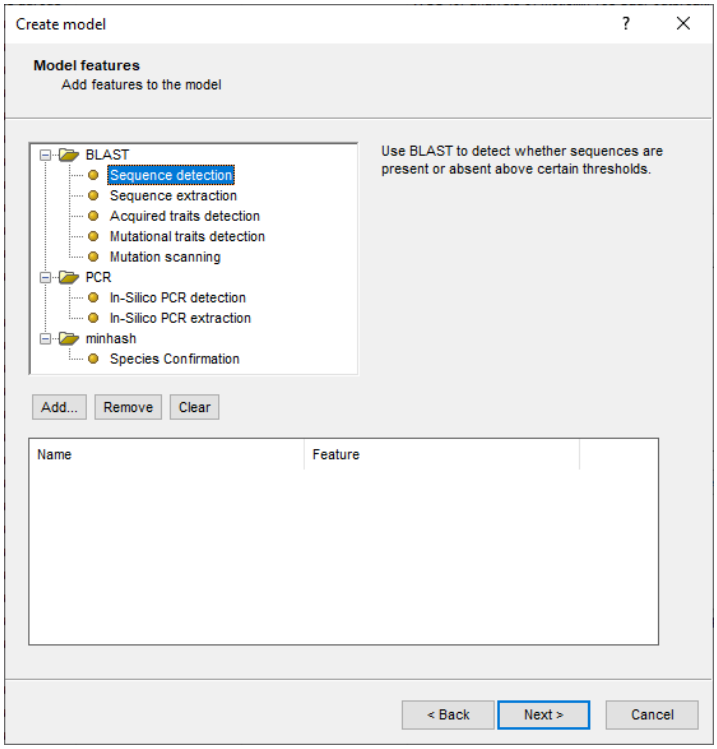

Figure 3.3: The *Create model* wizard, page 2

In the second page of the *Create model* wizard, one or more features can be added to the genotyping model. Note that a model should contain at least one feature.

In the tree control, highlight the type of feature (see 3.1 for more information) that you wish to add and press the <***Add...***> button.

The *New genotyping feature name* dialog box prompts you to enter a name for the new genotyping feature. When the <***OK***> button is pressed, the new feature will be listed in the bottom part of the *Create model* wizard.

As mentioned above, a genotyping model can contain more than one feature. Press the <***Add...***> button and provide a feature name if an additional feature is needed.

When a feature was added to the model in mistake, highlight the feature and press <***Remove***> to remove it from the list. Pressing <***Clear***> deletes all features at once from the model.

When the <***Next...***> button is pressed, the *Create feature* wizard (see Figure 3.4) will be started for each feature in the model.

The ***Knowledgebases*** section is available for all genotyping features. Press <***Change...***> to display the *Manage knowledge bases* dialog box, from which a knowledge base can be specified (see 2.3 for more information). When a knowledge base is selected, its ***Name*** and ***Version*** are indicated.

The ***BLAST*** section is available for all features using BLAST, i.e. sequence detection, sequence extraction, acquired traits detection, mutational traits detection and mutation scanning features.

In the ***BLAST*** panel, two settings for the BLAST algorithm can be specified:

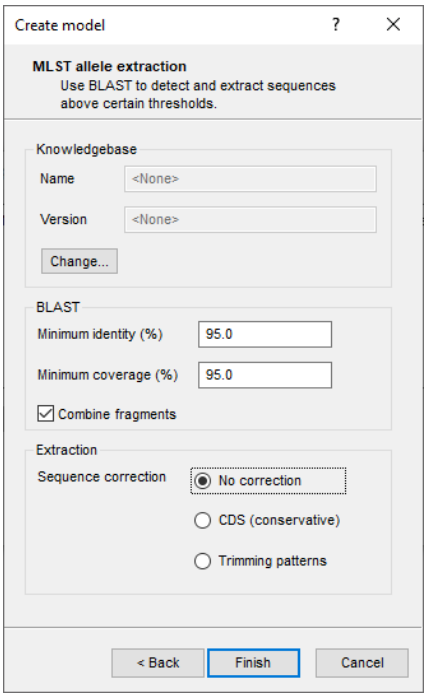

Figure 3.4: The *Create feature* wizard for a sequence extraction feature. For other features, the dialog box displays only a subset of the options displayed here.

- ***Minimum identity (%)*** is the minimum sequence identity (as percentage) of the query sequence against the knowledge base's reference sequences.
- ***Minimum length for coverage*** specifies the minimum overlap (as percentage) between the subsequence found in the target assembly sequence and the reference sequence from the knowledge base.

If the option ***Combine fragments*** is checked, genes that occur fragmented in the genome (i.e. split over two contigs) can still be detected.

The ***Extraction*** section is only available for BLAST-based sequence extraction features. When mismatches occur at the edges of a query sequence, BLAST may return a truncated sequence to optimize the similarity score. Therefore, when the knowledge base contains only a single allele or a very limited set of alleles for a certain gene, a ***Sequence correction*** might be needed. For the latter, one out of three options should be selected:

- ***No correction***: The BLAST hit is taken as-is. This is the default option since no correction is needed when the knowledge base covers sufficient diversity.
- ***CDS (conservative)***: The BLAST hit is extended to retrieve a full protein coding sequence (CDS), i.e. starting from the first encountered start codon upstream and ending at the first encountered stop codon downstream.
- ***Trimming patterns***: The BLAST hit is extended and trimmed to length using the trimming patterns, present in the knowledge base. These trimming patterns are similar to those used in batch assembly of Sanger sequences (see the Reference manual, Chapter Setting up sequence type experiments).

Pressing <***Finish***> will complete the creation of the feature in the model.

The *Create feature* wizard runs for each feature in the model. When the model creation is complete, the question ”Do you want to set the new model as active?” pops up. In the *Custom genotyping plugin*, only the active model can be executed, so typically you will want to answer <***Yes***> to this question.

Next, the software asks ”Do you want to modify the new model settings now?”. Pressing <***Yes***> will open the *Settings* dialog box, which is discussed in 4.

With <***Set as active***>, the highlighted model in the list will be set as the active genotyping model.

3.3 Updating a genotyping model

It is not possible to modify the knowledge base and other settings of an existing genotyping model. This is deliberately done to avoid any confusion regarding the settings used on entries analyzed earlier. When new insights require changes to the settings, the genotyping model should be *updated* so that these changes are reflected in a higher version of the same genotyping model or in a different genotyping model (by changing the name).

Therefore, the combination of model name and version unambiguously determine a genotyping model and this information is mentioned in the genotyping report (see 6.4).

Select ***Genotyping*** > ***Manage models...*** to display the *Manage genotyping models* dialog box.

Highlight the genotyping model that you want to update in the list and press <***Create update...***>. This action opens the *Create model* wizard with the info from the existing model.

Change the ***Model name*** and / or ***Version*** of the model to enable changing of the settings and press <***Next***>.

The procedure for creating an update is exactly the same as for creating a new genotyping model (see 3.2), except that the settings of the model on which the update is based are pre-filled in the wizard.

3.4 Extracting knowledge bases from a genotyping model

To aid in exchanging genotyping models between users, the knowledge base of a genotyping model in the BIONUMERICS database (referred to as an *embedded* knowledge base) can be exported to files.

Select ***Genotyping*** > ***Manage models...*** to display the *Manage genotyping models* dialog box. In this dialog, press <***Extract knowledgebases...***> to open the *Extract embedded knowledge bases* dialog box (see Figure 3.5).

Press <***Browse...***> to browse for an ***Exports directory***. Pressing <***Extract***> in the *Extract embedded knowledge bases* dialog box will export all selected embedded knowledge bases to the specified directory. A message box indicates that the export is completed. If the check box next to ***Open after export*** was checked the specified export directory will open.

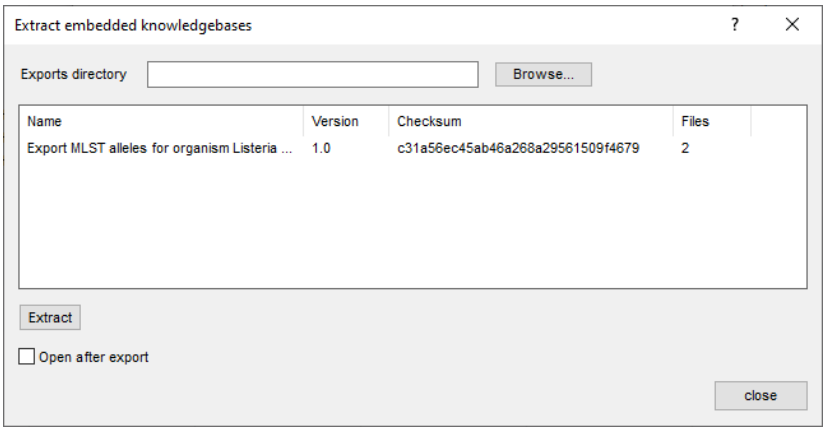

Figure 3.5: The *Extract embedded knowledge bases* dialog box.

3.5 Sharing genotyping models between databases

The highlighted genotyping model in the *Manage genotyping models* dialog box can be exported with <***Export...***>. This action opens the *Export genotyping model* dialog box (see Figure 3.6).

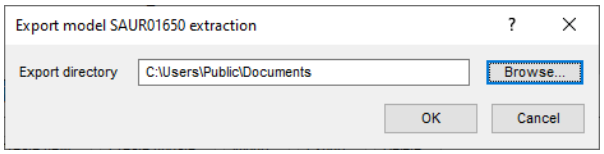

Figure 3.6: The *Export genotyping model* dialog box.

Browse for a location for the `*.bngtm` file, containing the genotyping model. Pressing <***OK***> will export the file and automatically open the export directory in Windows explorer.

To import one or more genotyping models from `*.bngtm` files, press the <***Import...***> button in the *Manage genotyping models* dialog box. This action will open the *Select model files* dialog box (see Figure 3.7).

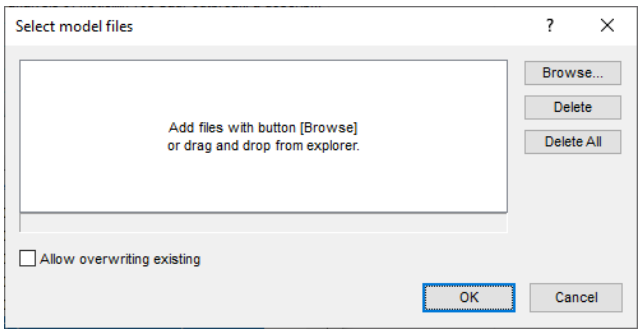

Figure 3.7: The *Select model files* dialog box.

Browse for the genotyping model file(s) to import. If ***Allow overwriting existing***, existing genotyping models in the database with the same name and version will be overwritten during the import. Pressing <***OK***> will import the models.

Chapter 4

Settings of the active genotyping model

4.1 Accessing the genotyping model settings

Settings for the active genotyping model in the *Custom genotyping plugin* can be accessed via ***Genotyping*** > ***Settings...*** in the *Main* window.

4.2 General settings

The *General* tab of the *Settings* dialog box (see Figure 4.1) holds settings for the genotyping reports and for general processing.

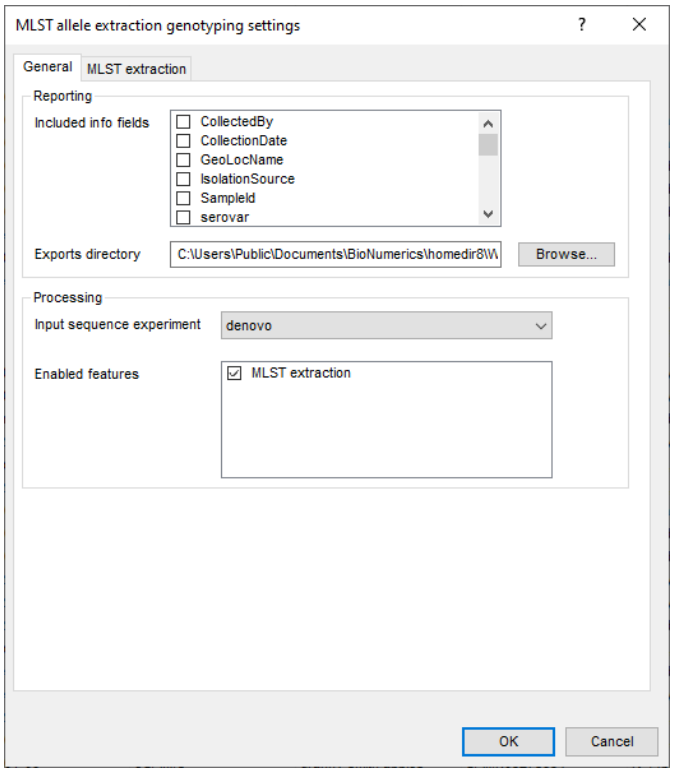

Figure 4.1: The *Settings* dialog box, *General* tab.

Under ***Reporting***, the entry information fields that will be displayed in the genotyping reports can

be specified in the ***Included info fields*** list. Simply check the ballot box next to an information field name to include the field in the report.

The ***Exports directory*** can be specified for all exports from the genotyping reports. By default, the exported files are stored in a subdirectory of the database directory, but a different location can be selected via the <***Browse***> button or entered directly in the text box.

The ***Input sequence experiment***, i.e. the sequence experiment containing the whole genome sequences to be screened, should be selected from the corresponding drop-down list. Select the <***Create***> option in case you wish to create a new sequence experiment type. In the latter case, make sure to import whole genome sequences in this experiment type before running the plugin.

It is crucial to specify at least the ***Input sequence experiment*** in the settings. If not specified, the error message "The input sequence experiment must be set to process entries." will be generated when the plugin is run.

In the ***Enabled features*** list, all features offered by the plugin are listed and enabled by default. If specific analyses are not required, you can uncheck them here to save on processing time and to omit the corresponding sections from the reports.

4.3 Feature settings

The *Settings* dialog box contains an additional tab for each feature defined in the active model. As an example, Figure 4.2 shows a tab for a sequence extraction feature called **MLST extraction**.

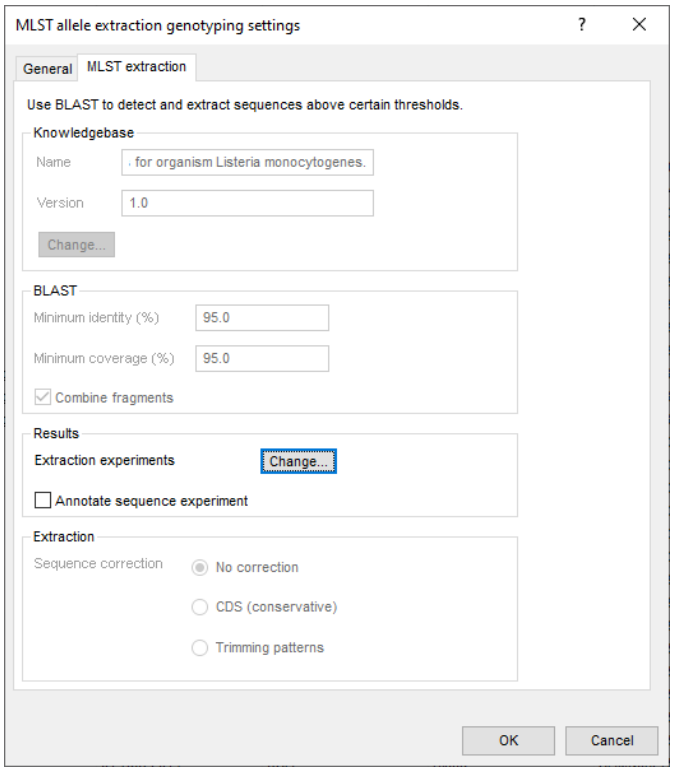

Figure 4.2: The *Settings* dialog box, showing the settings for a sequence extraction feature.

Most of the settings (e.g. the knowledge base used) in this tab will be read-only, since they are specified in the genotyping model. To change these, a new (see 3.2) or updated (see 3.3) genotyping model should be created and specified as the active model.

In case sequences are extracted (either using BLAST or via in silico PCR), it can be specified where the resulting sequences are stored.

Press the <***Change...***> button next to ***Extraction experiments*** to open the *Change sequence experiment* dialog box.

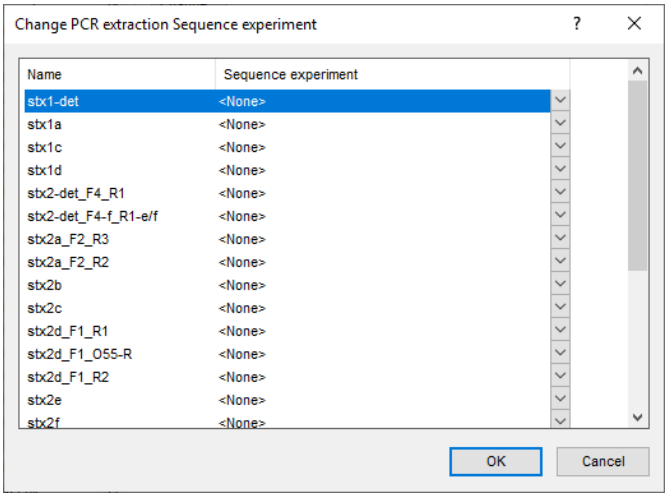

Figure 4.3: The *Change sequence experiment* dialog box.

The column 'Name' in the dialog contains all targets available in the knowledge base. Via the drop-down list in the 'Sequence experiment' column, an existing sequence experiment type can be specified in which the extracted sequence will be stored. When the <***Create***> option is selected, a dialog will pop up asking for the name of the sequence experiment type. By default, the name of the target will be suggested.

Check ***Annotate sequence experiment*** to annotate the input sequence with the detected genotyping features.

For species confirmation features, a ***Species confirmation info field*** can optionally be specified. This will contain the identification result, i.e. the name of the best matching reference in the species confirmation knowledge base. Depending on the score, this may be the genus name, the genus and species name or genus, species and subspecies name.

Chapter 5

Running the active genotyping model

5.1 Selecting entries

Once the plugin is installed and the settings have been specified, the actual screening of the genome sequences of the selected entries is an easy process.

Analyses are performed on the selected entries in the database. For example, to select a single entry, hold the **Ctrl**-key and click on the entry in the *Database entries* panel. Alternatively, use the **space bar** or click the ballot box next to the entry. In order to select a range of entries, hold the **Shift**-key and click on the last entry in the range.

More options for selecting entries can be found in the BIONUMERICS reference manual (see the Reference manual, Chapter Database entries).

5.2 Starting an analysis

Screening selected entries with the active genotyping model can be done using ***Genotyping*** $>$ ***Run active***.

The analysis time increases proportionally with the number of selected entries and the number of genotyping features in the model. A complete analysis may take up to several minutes or even hours. The progress bar disappears when the analysis is finished.

Chapter 6

Custom genotyping reports

6.1 Opening custom genotyping reports

A custom genotyping report (see Figure 6.1) can be opened for the selected entries with ***Genotyping*** > ***Reports...***.

Clicking on an entry in the *Entries* panel of the *Genotyping report* window (or using the up and down arrow keys on the keyboard) shows the report for the highlighted entry.

At the top of each report the creation date of the report (***Date***), the Key (***Name***), and information fields that were checked in the *General* tab of the genotyping settings are displayed, followed by a summary of the results of all analyzed traits.

Selecting ***File*** > ***Exit*** closes the *Genotyping report* window.

6.2 Report styles

In the *Genotyping report* window, three different report styles can be applied from the drop-down list in the panel header or via the menu (***Report*** > ***Report styles***):

1. ***Summary***: only a summary of the results is shown.
2. ***Default***: the summarized results and most details are shown in a tabular format. In this report style, all columns of the results tables can be sorted alphabetically or numerically by clicking on their headers.
3. ***Complete***: the summarized results and all available details are shown. More exhaustive information is presented in an additional row, for example descriptions of the detected genes, decision trees, etc.. Result tables cannot be sorted in this report style.

6.3 Details section

6.3.1 Introduction

The Details section in the genotyping report contains a detailed result table for each analyzed genotyping feature.

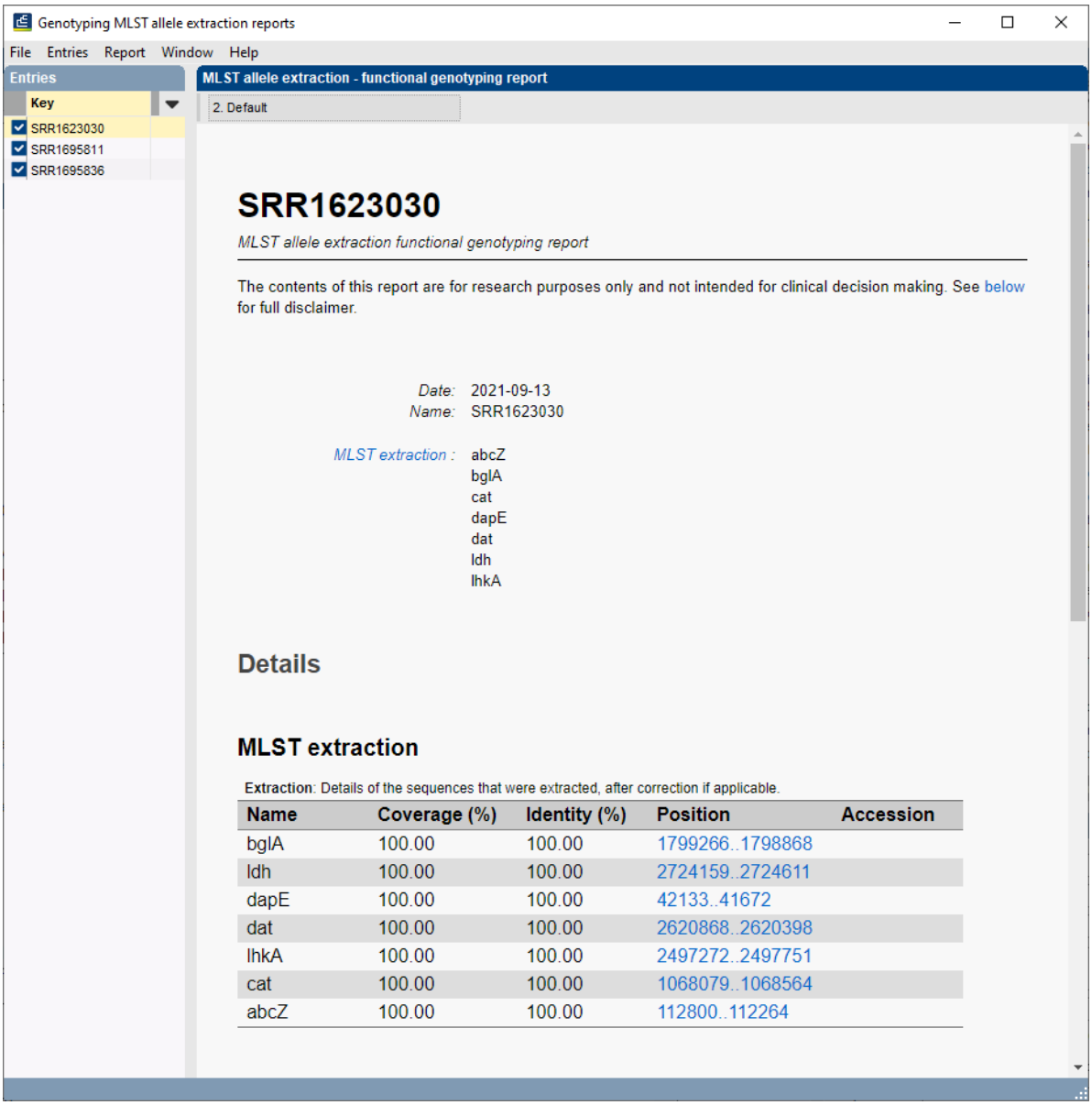

Figure 6.1: The *Genotyping report* window, showing a report extraction of MLST alleles.

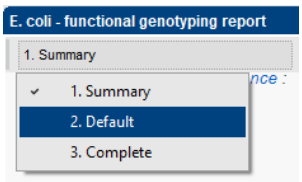

Figure 6.2: Drop-down list to select report styles in the *Genotyping report* window.

6.3.2 BLAST-based features

All detected sequences or traits are listed in the table, with following fields:

- **Identifier:** Unique ID of the detected sequence. Note that in the trait detection result table this field is only visible in the ***Complete*** report style of the report.

- **Trait:** Trait linked to the detected sequence. Multiple traits can be linked to the same sequence.
- **Name:** Name of the detected sequence.
- **Coverage (%):** Percent overlap between the query and the target sequence.
- **Identity (%):** BLAST identity, expressed as a percentage.
- **Position:** Position(s) of the BLAST match on the input genome. Multiple positions are possible when ***Combine fragments*** was checked in the genotyping settings. This field contains hyperlinks which open the *Sequence editor* window in BIONUMERICS with the positions highlighted on the sequence.
- **Accession:** This is an optional field to store the accession corresponding to the detected sequence. GenBank and RefSeq accessions are automatically hyperlinked in the report.
- **Description:** This is an optional field to store additional information of the reference sequence. If this field is present in the knowledge base, the field can be displayed in the ***Complete*** template of the report.
- **Publication:** This is an optional field to store the publication source of the sequence. PubMed, PubMed Central and DOI identifiers are automatically hyperlinked in the report. If this field is present in the knowledge base, the field can be displayed in the ***Complete*** template of the report.

6.3.3 PCR-based features

All detected PCR targets are listed in the table, with following fields:

- **Identifier:** ID of the PCR target.
- **Position:** Position on the input sequence where the target was detected. This field contains hyperlinks which open the *Sequence editor* window in BIONUMERICS with the position highlighted on the sequence.
- **Strand:** The strand on which the PCR primers were detected: either the base strand ('base') or the reverse-complement strand ('revcomp').
- **Length:** Length in nucleotides of the in silico PCR product.
- **Reference length:** Expected length of the target sequence, including the primers themselves.

6.3.4 Minhash-based features

The best matching identification with one of the reference genomes from the knowledge base is listed in the table, with following fields:

- **Confirmation:** The species confirmation (i.e. identification) result, at genus, species or subspecies level.
- **Containment score (%):** Sourmash containment score, expressed as a percentage.

- **Largest exceeded threshold (%):** The highest containment score threshold (as defined in the knowledge base) that was exceeded.
- **Ref. genome name:** Organism name of the reference genome as obtained from GenBank.
- **Accession:** GenBank accession of the reference genome sequence.
- **Taxid:** Taxonomy ID as used in NCBI's taxonomy browser (https://www.ncbi.nlm.nih.gov/Taxonomy/Browser/wwwtax.cgi).

6.4 Info section

At the bottom of each report, an *Info section* is shown which contains information regarding the analysis date, plugin version, knowledge base name and version, and settings of each analysis.

6.5 Exporting report information

The genotype information for all selected entries in the *Genotyping report* window can be exported with ***Entries*** > ***Export selected***. The results for the currently shown report can be exported with ***Report*** > ***Export current***.

A Tab Separated Values (`*.tsv`) file is created for each functionality and stored in the report export directory as specified in the genotyping settings. The location of the files is opened after the export.

The displayed report can be printed directly to a printer using ***Report*** > ***Print...***.

The dialog box that appears is the standard Windows Print dialog box, allowing you to choose a printer and change the properties. Depending on your system, this also allows printing to a PDF file to create an export of the displayed report. Please note that background colors, such as those in lists and mutational decision trees, may be lost in this step regardless of printer settings.

Chapter 7

Tools for generating knowledge base files

7.1 Introduction

The *Custom genotyping plugin* offers tools to facilitate the creation of knowledge bases (see 2). This requires that the knowledge base data is stored in a BIONUMERICS database. This database may be different from the database in which the final knowledge is actually used.

7.2 Exporting Minhash signatures

A tool is available to create the sourmash signatures file (see 2.2.9) in the correct format for use in a species confirmation knowledge base.

Select the entries with reference genomes to be included in the knowledge base and use ***Genotyping*** > ***Tools*** > ***Export Minhash signatures...***. This action opens the *Export Minhash signatures* dialog box (see Figure 7.1).

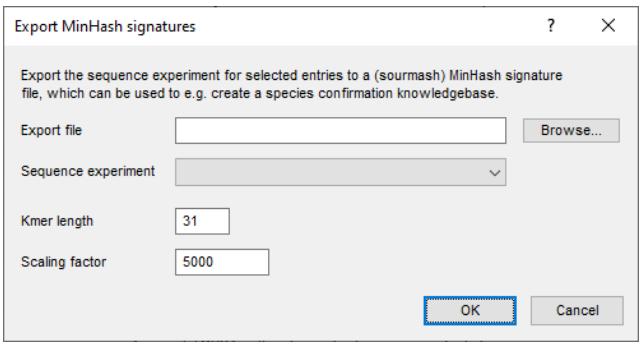

Figure 7.1: The *Export Minhash signatures* dialog box.

Browse for an ***Export file*** (`*.sig`) to export sourmash signatures to for the genome sequences stored in the specified ***Sequence experiment***. Any filename can be specified, but for use in a species confirmation knowledge base it should be named `genomes.sig`.

The sourmash parameters ***Kmer length*** (default value: 31) and ***Scaling factor*** (default value: 5000) can be specified. For more information, see the Reference manual, Chapter Minhashing based cluster analysis of sequences. When used in a species confirmation knowledge base, the

actual values employed for these parameters should be copied to the sourmash parameters file (see 2.2.8).

Press <***OK***> to write the JSON formatted signatures file. Each signature is labeled with the entry key as `entry_key.fasta`.

7.3 Exporting knowledge base FASTA files

A tool is available for exporting sequences to FASTA formatted files in predefined configurations used in various genotyping knowledge bases.

Select the entries with genomes to be included in a knowledge base FASTA file and use ***Genotyping*** > ***Tools*** > ***Export knowledgebase fasta...***. This action opens the *Export knowledge base fasta* dialog box (see Figure 7.2).

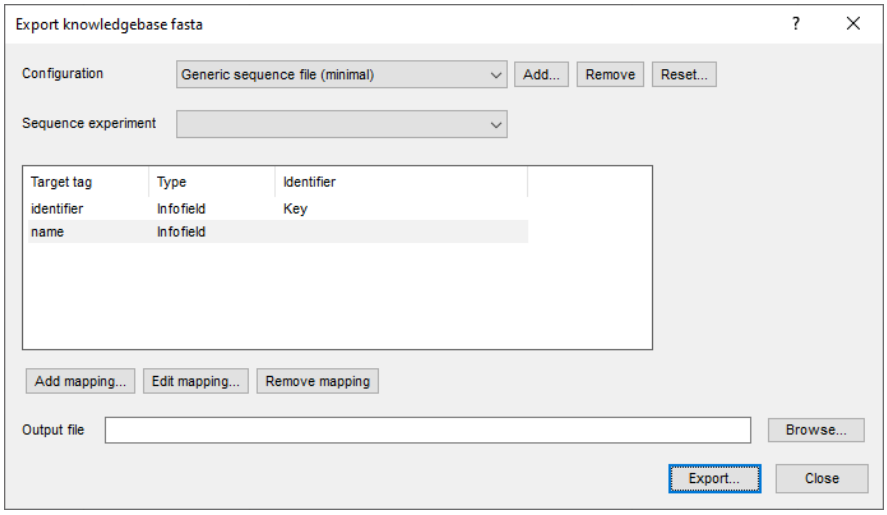

Figure 7.2: The *Export knowledge base fasta* dialog box.

Firstly, a ***Configuration*** should be specified via the corresponding drop-down list. Extra configurations can be added to the list (<***Add...***>) or removed when not needed (<***Remove...***>). Pressing <***Reset...***> will reset the list with configurations to its factory defaults, removing all customizations and mappings. By default, following (empty) configurations are present:

- Generic sequence file (minimal)
- Generic sequence file (standard)
- Generic sequence file (standard + traits)
- Acquired resistance
- Mutational resistance (loci)
- Acquired virulence
- Virulence islands
- Phage detection
- Serotype (Listeria)

- Serotype (E. coli)
- Plasmid detection (Ori) (E. coli)
- Acquired virulence (E. coli)
- Resistance type (AA/NA, do not combine) (E. coli)

The ***Sequence experiment*** containing the genome sequences can be picked from the corresponding drop-down list.

For the selected configuration, a mapping between FASTA header fields and information available in the BIONUMERICS database (entry information field or characters) should be made. A new mapping can be created by pressing <***Add mapping...***>. This action opens the *Create mapping* dialog box (see Figure 7.3).

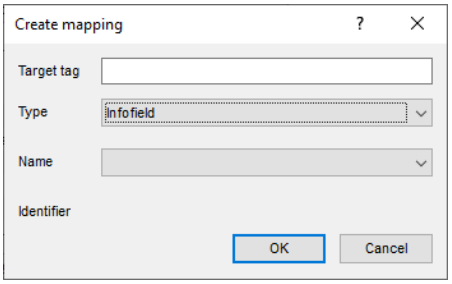

Figure 7.3: The *Create mapping* dialog box.

The ***Target tag*** is the tag used in the FASTA header line.

Type is either 'Infofield' or 'Characters'. With 'Infofield' specified, an entry information field name can be selected as ***Name***. With 'Characters' specified, a character type experiment can be selected from the ***Name*** drop-down list.

Identifier shows the information field ID or the character experiment ID, respectively, which might be different from the display name.

When editing an existing mapping, ***Target tag*** and ***Type*** will be read-only.

Pressing <***OK***> will create the mapping.

The highlighted mapping in the list can be edited with <***Edit mapping...***>, which will open the same *Create mapping* dialog box. <***Remove mapping***> removes the highlighted mapping.

When all required target tags in the configuration are mapped, browse for an ***Output file*** and press <***OK***>. BIONUMERICS will ask for confirmation before actually exporting the FASTA file.

Bibliography

[1] C Titus Brown and Luiz Irber. sourmash: a library for minhash sketching of dna. *Journal of Open Source Software*, 1(5):27, 2016.

bioMérieux S.A. • 69280 Marcy l'Étoile • France • Tel.: + 33 (0)4 78 87 20 00 • Fax: +33 (0)4 78 87 20 90
www.biomerieux.com