Sherlock v1.0

AUTOMATED PROCESS PIPELINE FOR STATISTICAL EVALUATION OF PATRISTIC TREE-DISTANCE VARIATION

NATHAN SEIDEL | 2019

Summary

Sherlock is an automated process pipeline for statistical evaluation of patristic tree-distance variation given partitioned or non-partitioned molecular sequence data. Patristic distances are a measure for phylogenetic diversity in data sets (Faith, 1992), but the patristic distance measures do not reveal the impact of statistical or other biases within molecular data; i.e. if the patristic distances inferred from sequences in a data set are randomly distributed and thus biased. Therefore, Sherlock randomly samples sequences as random data sets to compare the original patristic distances to a random distribution of comparable data sets.

Sherlock re-aligns the original and randomly sampled sequence pools with one of several MAFFT implemented alignment algorithms (Katoh & Standley, 2013). For each alignment Sherlock conducts a phylogenetic reconstruction using either FastTree (Price, Dehal, & Arkin, 2010) or IQ-TREE (Nguyen, Schmidt, von Haeseler, & Minh, 2015). Branch lengths of phylogenetic trees (patristic distances) are analysed statistically, whereby inferred patristic distances of original data trees are compared with sampled distances of sub-pool related random data trees (see 5. Workflow of Sherlock) after resolution of polytomies with ape (Paradis & Schliep, 2018).

The pipeline accepts interleaved and non-interleaved fasta formatted sequence files in a specifically set up directory structure (see 3.2. Folder structure). The directory structure reflects a total sequence pool from which Sherlock arbitrarily samples user-defined numbers of sequences to create random sub-pools.

The number of sequences, distribution of categories (e.g., the species of species assigned sequences) and the number of sub-pools (i.e. repetitions of random sampling) follow user-defined specifications (see 3.3. Parameter file (.tsv)).

The pipeline generates a histogram and density distribution of inferred patristic distances of each randomly generated sub-pool with ggplot2 (Wickham, 2016) and gridExtra (Auguie, 2017). For direct comparison, the inferred patristic distance of the corresponding original data set is graphically highlighted in each distribution of random sub-pool patristic distances (see 5.5.1. Histograms and distribution). Additionally, patristic distances of randomised sub-pools are grouped in violin plots with ggplot2 (Wickham, 2016) and gridExtra (Auguie, 2017) according to number of sequences and number of categories (see 5.5.2. Violin plots).

Besides graphical output, Sherlock generates informative text files, e.g. sequences occurring multiple times or an off-range list of sub-pools, when an original patristic distance lies not within the random distribution of sub-pool patristic distances. Patristic distances for all original data sets and sub-pools are summarised in text files, as well (see 5.5.3. Other information and results).

Sherlock was developed under Ubuntu in PERL v5.26.1 and R v3.4.4 and runs on Linux operating systems. The pipeline is started directly from the command line in terminal. For further details about Sherlock usage, please see 4. Run Sherlock.

Table of contents

Table of figures

1. Contents

Sherlock consists of the following components; all of the following files have to be stored in the same folder:

1.1. run_Sherlock.pl

The executable PERL script which controls the pipeline processes.

1.2. Sherlock.pm

A PERL module which contains all main script functions to generate the randomised data sets and run the analyses of patristic tree distance variation.

1.3. resolvePolytomies.R

Resolves conflicting internal nodes dichotomously in polytomous calculated trees using functions from ape (Paradis & Schliep, 2018).

1.4. ggplotGraphicsOutput.R

An R script graphically processing each randomised data set utilising functions from ggplot2 (Wickham, 2016) and gridExtra (Auguie, 2017). For each randomised data set, three histograms and density distributions for inferred branch lengths are plotted as total, median and mean. All single data set PDF are merged into one PDF, too.

1.5. ggplotAllViolin.R

Plots the inferred patristic distances as violin plots according to 1) the number of sequences and 2) the number of categories (e.g. species) utilising functions from ggplot2 (Wickham, 2016) and gridExtra (Auguie, 2017).

2. Installation

2.1. External programs

Sherlock allows to perform analyses on patristic distance variation in sequence data sets. Hence the following external programs are necessary in order to run Sherlock properly:

- **MAFFT** (for more information, please visit: https://mafft.cbrc.jp/alignment/software/)
- **FastTree** (for more information, please visit: http://www.microbesonline.org/fasttree/)
- **IQ-TREE** (for more information, please visit: http://www.iqtree.org/)
- **R** (for more information, please visit: https://www.r-project.org/)
    - ape (for more information: https://www.ape-package.ird.fr/)
    - ggplot2 (for more information: https://www.ggplot2.tidyverse.org/)
    - gridExtra (for more information: http://CRAN.R-project.org/package=gridExtra)
    - e1071 (for more information: https://CRAN.R-project.org/package=e1071)

All of these are mandatory – except for FastTree and IQ-TREE. It is possible to choose only (but at least) one tree reconstruction program.

2.2. Installation options

Sherlock contains an installation file install.sh to install Sherlock and all or some of the external software. Usage of install.sh: `install.sh <options>`

Possible options are:

all	: install all components (mafft, fasttree, iqtree, R-full)
mafft	: install mafft
fasttree	: install fasttree
iqtree	: install iqtree
R-full	: install r-base, r-recommended, ape, ggplot2, gridExtra, e1071
R-packages	: install R-packages ape, ggplot2, gridExtra, e1071
ape	: install R-package ape (v5.3)
ggplot	: install R-package ggplot2 (v3.1.0)
gridExtra	: install R-package gridExtra (v2.3)
e1071	: install R-package e1071 (v1.7.1)

If an option is used and one or more of the packages are already installed on the machine, the newest available version from apt-get is installed or the version downloaded by the installer. If you do not want to overwrite existing installations, choose which components you wish to install.

For example, to install all components install.sh is called as follows:

```
sudo bash install.sh all
```

If MAFFT and R are pre-installed, but neither FastTree nor IQ-TREE and none of the R-packages:

```
sudo bash install.sh fasttree iqtree R-packages
```

If all components excluding R-packages ape and e1071 are installed:

```
sudo bash install.sh ape e1071
```

2.3. Download and install Sherlock

To install Sherlock, download the program from https://github.com/NathanSeidel/Sherlock and extract the downloaded file *Sherlock-master.zip*. Compressed files and folders (e.g. .zip or .tar.xz) can be extracted via right-click or via terminal.

To extract Sherlock, open a terminal (press Ctrl-Alt + T) and navigate to the location where Sherlock was downloaded to (e.g. `~/Downloads/`) and type

```
user@user:~/Downloads$ unzip Sherlock-master.zip
```
<enter>

The contents of the compressed folder are extracted to `Sherlock-master/` (user is the name of the local login). To extract the main program, type

```
user@user:~/Downloads$ tar -xf Sherlock-master/Sherlock.tar.xz
```
<enter>

All necessary installation files are extracted to `~/Downloads/Sherlock-master/Sherlock/`. To run Sherlock, some external software must be installed (see 2.1. External programs).

If you want to install Sherlock and all of the external components (see 2.2. Installation options), type

```
user@user:~/Downloads$ sudo bash Sherlock-master/Sherlock/install.sh all
```
<enter>

Before the installation, the installer asks you, if you want to conduct changes to the system and install Sherlock and its' dependencies. If the question is confirmed (by typing 'yes'), Sherlock proceeds to install Sherlock and the aforementioned external programs you choose. If they are already installed on the machine, the newest available version is installed from apt-get or the version downloaded by the installer (see 2.1. External programs). Existing installations are overwritten.

For other installation options, see 2.2. Installation options.

After the installation, Sherlock now can be used from any location on the machine and the folder `Sherlock/` is deleted.

3. Preparation of data

3.1. Fasta file naming convention

The random generation of random sequence data sets in Sherlock is not entirely random, because the sequences are sampled according to categories. Categories can be any keyword encoded in the sequence names or identifiers. Thus, keywords or words within a sequence name must be delimited by underscores.

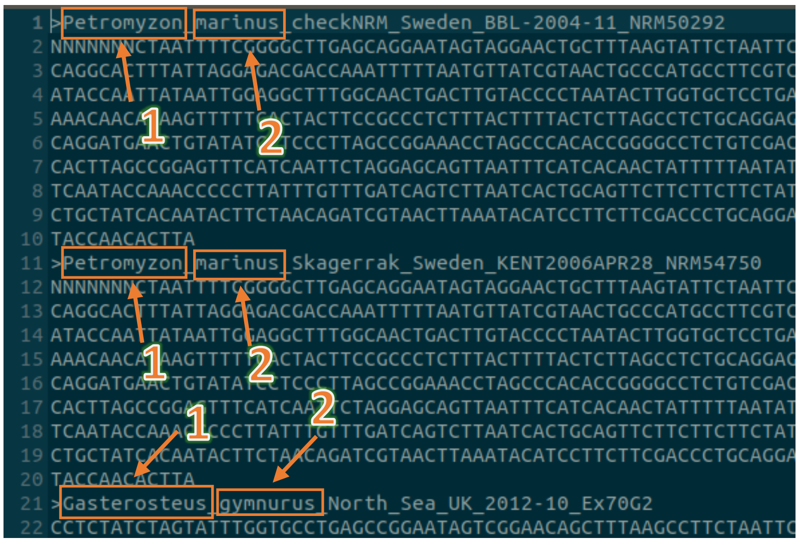
For example, a sequence contains information on the species of the specimen, the location and the country:

Sequence name	*>Genus_species_subspecies_location_country*				
Keywords	Genus	species	subspecies	location	country
Keyword positions	1	2	3	4	5

Fig. 1: Explanation of keywords and keyword positions in sequence names to categorise sequences.

What information is encoded in the sequence name is not restricted by any means. Each of the underscore-delimited words can be a keyword to group sequences according to a category. Therefore, the order and position of the words within the sequence name is important.

In the previous example (Fig. 1), the category “Genus” is at position 1, category “species” is at position 2 and so on. This order must be consistent for all species names and identifiers, because Sherlock accepts positional information on which sequences are grouped together.



```
>Petromyzon_marinus_checkNRM_Sweden_BBL-2004-11_NRM50292
NNNNNNNCTAATTTTCGGGGCTTGAGCAGGAATAGTAGGAACTGCTTTAAGTATTCTAATTC
CAGGCACTTTATTAGGAGACGACCAAATTTTTAATGTTATCGTAACTGCCCATGCCTTCGTC
ATACCAATTATAATTGGAGGCTTTGGCAACTGACTTGTACCCCTAATACTTGGTGCTCCTGA
AAACAACATAAGTTTTTGACTACTTCCGCCCTCTTTACTTTTACTCTTAGCCTCTGCAGGAG
CAGGATGAACTGTATATCCTCCCTTAGCCGGAAACCTAGCCCACACCGGGGCCTCTGTCGAC
CACTTAGCCGGAGTTTCATCAATTCTAGGAGCAGTTAATTTCATCACAACTATTTTTAATAT
TCAATACCAAACCCCCTTATTTGTTTGATCAGTCTTAATCACTGCAGTTCTTCTTCTTCTAT
CTGCTATCACAATACTTCTAACAGATCGTAACTTAAATACATCCTTCTTCGACCCTGCAGGA
TACCAACACTTA
>Petromyzon_marinus_Skagerrak_Sweden_KENT2006APR28_NRM54750
NNNNNNNCTAATTTTCGGGGCTTGAGCAGGAATAGTAGGAACTGCTTTAAGTATTCTAATTC
CAGGCACTTTATTAGGAGACGACCAAATTTTTAATGTTATCGTAACTGCCCATGCCTTCGTC
ATACCAATTATAATTGGAGGCTTTGGCAACTGACTTGTACCCCTAATACTTGGTGCTCCTGA
AAACAACATAAGTTTTTGACTACTTCCGCCCTCTTTACTTTTACTCTTAGCCTCTGCAGGAG
CAGGATGAACTGTATATCCTCCCTTAGCCGGAAACCTAGCCCACACCGGGGCCTCTGTCGAC
CACTTAGCCGGAGTTTCATCAATTCTAGGAGCAGTTAATTTCATCACAACTATTTTTAATAT
TCAATACCAAACCCCCTTATTTGTTTGATCAGTCTTAATCACTGCAGTTCTTCTTCTTCTAT
CTGCTATCACAATACTTCTAACAGATCGTAACTTAAATACATCCTTCTTCGACCCTGCAGGA
TACCAACACTTA
>Gasterosteus_gymnurus_North_Sea_UK_2012-10_Ex70G2
CCTCTATCTAGTATTTGGTGCCTGAGCCGGAATAGTCGGAACAGCTTTAAGCCTTCTAATTC
```

Fig. 2: Excerpt from a sequence file highlighting keyword positions 1 and 2. All sequence names contain a genus at keyword position 1 and the species name at keyword position 2.

Sherlock accepts one or two positions to group and categorise sequences. The two positions are passed to Sherlock joined by ‘+’, e.g. passing 1+2 to Sherlock categorises sequences according to “Genus” and “species” (see Fig. 1 and Fig. 2).

It is important, that **all** sequence names are named according to the same order of categories. The content of positions is irrelevant: the category “Genus” can occur at any position in a sequence name, but all sequence names of the same data set have to be consistently named. Sherlock itself has no possibility to check, if a given position contains the same content in all sequence names.

3.2. Folder structure

To run Sherlock, input data must be stored in a specific directory hierarchy. So as to analyse the patristic distance variation in molecular sequence data sets, the data is assumed to be partitioned into sub-sets – e.g. different sampling sites of sequences, different sampling points in time, etc.

Each partition or sub-pool is part of a total sequence pool, e.g. a greater area with several sampling locations or samples collected during temporal measurement series. The partitions or sub-pools must consist of (at least) one or more sequence files in interleaved or non-interleaved fasta file format (.fa, .fas, .fasta).

Thus, the directory structure for an example data set must be structured as follows:

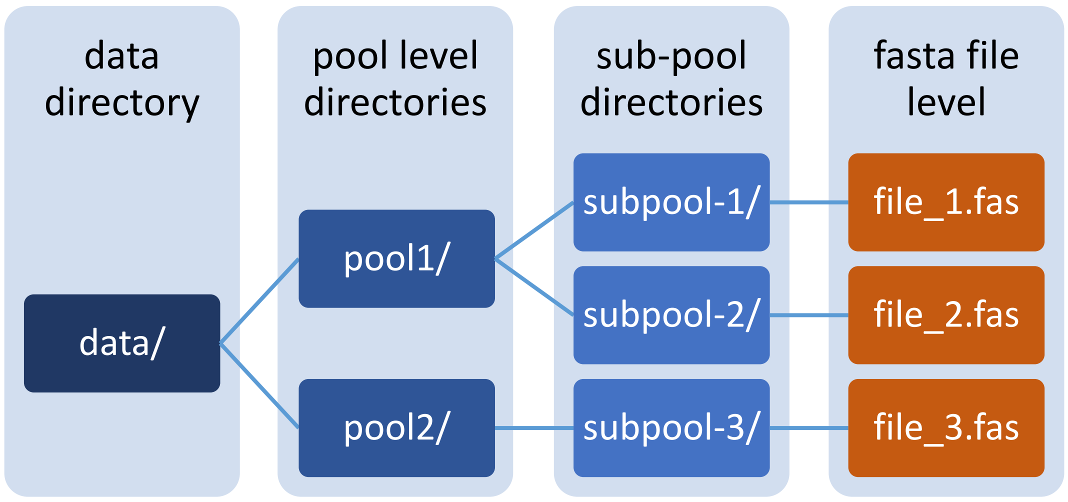


Fig. 3: Directory hierarchy for Sherlock input data. The pool directories are partitioned into sub-pools containing at least one fasta file. Sherlock allows to analyse several pools at once. Different pools don’t necessarily have to be present in the same parent directory (as depicted), but can be stored in different locations.

The sequences of the sub-pools form the total set of sequences in a pool. All sequences in such a pool are the basis for the generation of new random data sets. By randomly sampling a fixed number of sequences from a given number of categories from this total set of sequences, the impact of random processes on the original sub-pools can be estimated and analysed.

3.3. Parameter file (.tsv)

The parameter file defines how to analyse the sub-pools or subsets and how to generate the random data sets for each sub-pool or subset. The parameter file is processed line wise.

The first line is the head line:

RUN REPNR SPNR SEQNR POOL ALIGN TREE

3.3.1. RUN

RUN must be named the same as the name of an original sub-pool folder within a pool. According to the example in Fig. 2, possible run names are subpool-1, subpool-2 and subpool-3.

3.3.2. REPNR

REPNR must be a positive integer number specifying how many random data sets are generated for the current run.

3.3.3. SPNR

SPNR must be a positive integer number specifying how many categories are randomly sampled from the pool. By definition, each randomly sampled category occurs at least with one sequence in a generated random subset. SPNR corresponds to naming convention and given keyword positions (see 3.1. Fasta file naming convention).

3.3.4. SEQNR

SEQNR, like **SPNR**, must be a positive integer number and defines how many sequences will be randomly sampled from the randomly sampled categories.

By definition, the exact number of categories and sequences is sampled, if the number of sequences and categories for a keyword or keyword combination is sufficient. Otherwise, the line is not processed.

3.3.5. POOL

POOL is the full path to the pool from which the sequences are supposed to be drawn.

3.3.6. ALIGN

ALIGN defines the alignment algorithm the data is re-aligned with. Possible options are:

- mafft
- linsi
- einsi
- ginsi

3.3.7. TREE

TREE defines the tree reconstruction program to infer phylogenetic trees from the original and random data sets to calculate patristic distances. Possible options are:

- fasttree
- iqtree

These options are equivalent to the commands

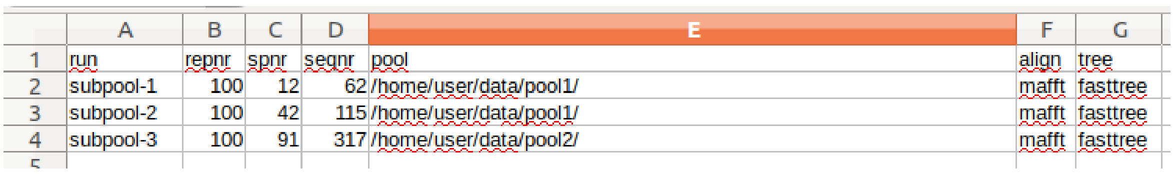
```
fasttree -gtr -nt -nosupport > file.nwk
```

and

```
iqtree -nt AUTO -keep-ident -m GTR -s file.aln
```

3.3.8. Example parameter file

In accordance to Fig. 2, a corresponding parameter file is structured as follows:



	A	B	C	D	E	F	G
1	run	repnr	spnr	seqnr	pool	align	tree
2	subpool-1	100	12	62	/home/user/data/pool1/	mafft	fasttree
3	subpool-2	100	42	115	/home/user/data/pool1/	mafft	fasttree
4	subpool-3	100	91	317	/home/user/data/pool2/	mafft	fasttree

Fig. 4: Screenshot of an exemplary parameter file based on Fig. 2. It is assumed that data is located in the user's home directory.

4. Run Sherlock

4.1. Start Sherlock via command line

Sherlock is started directly via command line command. After Sherlock is installed via install.sh (see 2. Installation), the program can be started from any directory or location on the machine.

Each command line to start Sherlock begins with "Sherlock", followed by the necessary arguments passed to Sherlock. The arguments must be passed in order for Sherlock to run correctly:

- Sherlock <parameter file> <output folder> <data directory> <naming convention>

The user specifies the location of the parameter file with/under which the analyses are conducted, the directory where the results are saved, the directory of the original data sets and a naming convention to categorise the sequences.

The first three arguments are locations of files or folders. They can either be passed as full paths or relative paths, depending on the directory Sherlock is started from. If the results folder already exists, its' contents are deleted. Otherwise the results directory will be created.

Naming convention must be passed as two numbers separated by '+' (see 3.1. Fasta file naming convention).

Sherlock asks at program start, if duplicates of sequences in partitions or sub-pools of a pool are allowed and – in case, duplicate sequence names occur – if Sherlock is supposed to proceed. If the question is confirmed with 'yes', Sherlock starts.

4.2. Example command line for Sherlock

The downloaded Sherlock files contain a folder "example" with an exemplary data set consisting of a data directory and a parameter file. To start Sherlock with the example data, move the folder to your machines home directory. Open a terminal prompt and type:

```
user@user:~$ Sherlock example/ex_paramf.tsv ex_results/ example/data/ 1+2
```

<enter>

Sherlock reads the parameter file located at `/home/user/example/` and writes results to `/home/user/ex_results/`. The data set located at `/home/user/example/data/` is input to the Sherlock pipeline. Sequences are categorised according to underscore-delimited keyword positions 1 and 2.

4.3. Options

Sherlock offers single options that can be invoked by prepending '--' to the option:

4.2.1. help

Invoking the help option shows usage information to start Sherlock:

```
user@user:~$ Sherlock --help <enter>
```

4.3.2. version

Prints the version, copyright and license information.

```
user@user:~$ Sherlock --version <enter>
```

4.3.3. citation

Invoking Sherlock with the citation options, Sherlock prints the citation for publications and a BibTeX citation:

```
user@user:~$ Sherlock --citation <enter>
```

4.3.4. uninstall

Invoking the uninstall option uninstalls Sherlock and the components installed during Sherlock installation. Use sudo to permit deinstallation:

```
user@user:~$ sudo Sherlock --uninstall <enter>
```

5. Workflow of Sherlock

5.1. Sherlock workflow

Sherlocks purpose is to analyse variation of patristic distances in molecular sequence data. Patristic distances are a measure for phylogenetic diversity in data set (Faith, 1992), but the patristic distance measures of total branch length (TBL), mean branch length (mean BL) and median branch length (median BL) in an inferred tree do not reveal the impact of statistical or other biases; i.e. if the patristic distances inferred from sequences sub-pool data sets is randomly distributed and thus biased.

Therefore, Sherlock calculates the patristic distances for the original sub-pool data sets and compares the results to the distribution of inferred patristic distances calculated from randomly sampled sub-pool data sets. The random sub-pool data sets best match the number of sequences and categories (e.g. species) in the original sub-pool data sets.

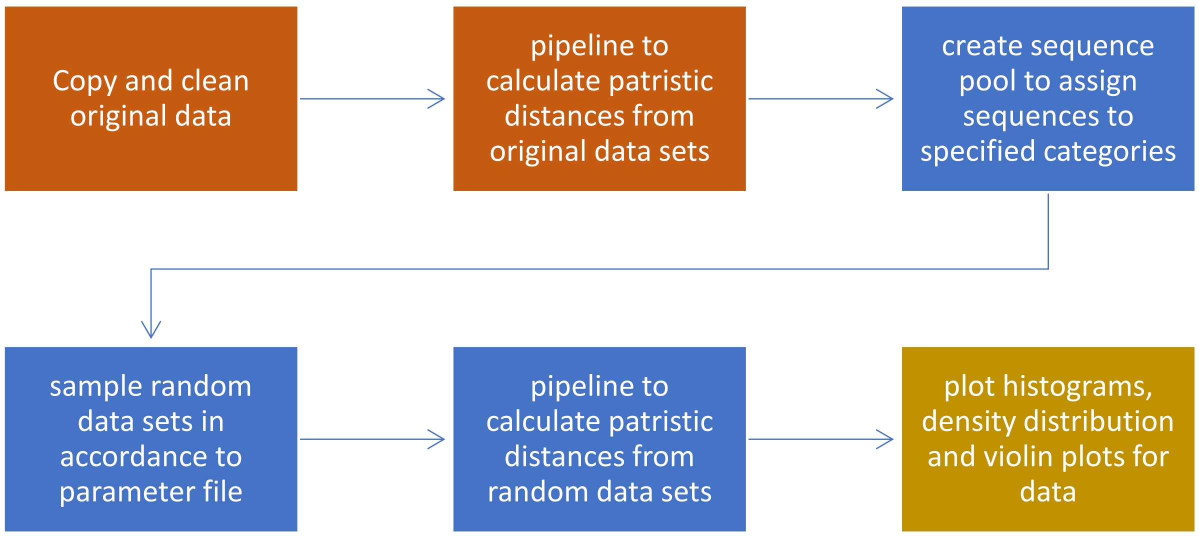

Copy and clean original data
pipeline to calculate patristic distances from original data sets
create sequence pool to assign sequences to specified categories
sample random data sets in accordance to parameter file
pipeline to calculate patristic distances from random data sets
plot histograms, density distribution and violin plots for data

Fig. 5: Workflow of Sherlock controlled pipeline.

5.2. Pipeline data preparation

First, the pipeline copies the original data sets specified in the parameter file (see 3.3. Parameter file (.tsv)) to the output folder. Gaps are removed from the copied sequence files. The cleaned sequence files are passed to the main pipeline to infer patristic distances (Fig. 6).

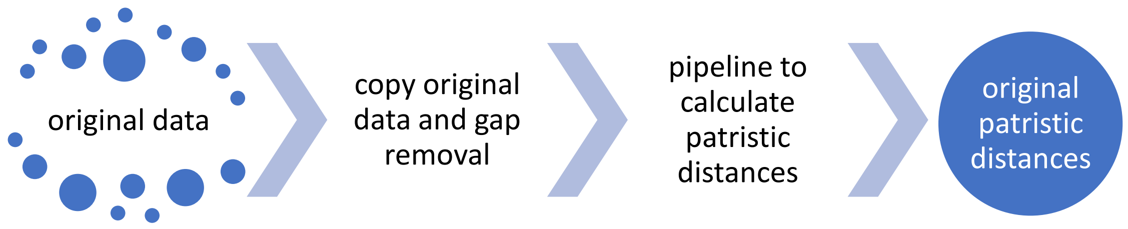

original data
copy original data and gap removal
pipeline to calculate patristic distances
original patristic distances

Fig. 6: Processes involved in preparation of data sets before pipeline processing.

5.3. Pipeline to calculate patristic distances

To calculate patristic distances, which are an indicator of phylogenetic diversity, all original sub-pool data sets are aligned with MAFFT (see 2.1. External programs). Afterwards, the alignments are passed to a tree reconstruction method (see 2.1. External programs) and possibly occurring polytomies within the calculated trees are resolved to create fully dichotomous trees. Resolved polytomies are assigned branch length zero.

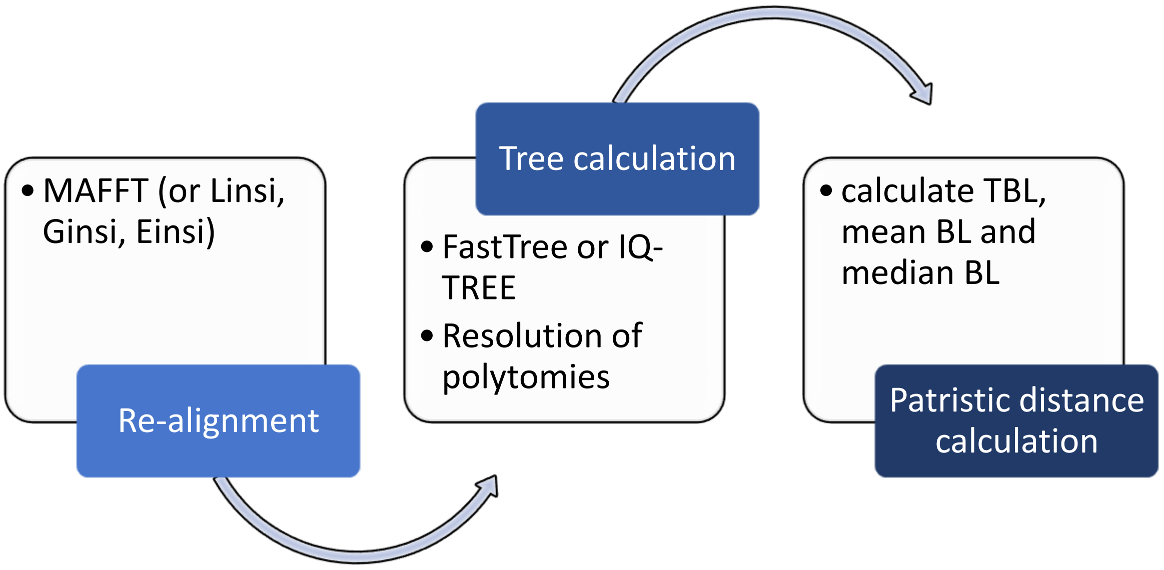

• MAFFT (or Linsi, Ginsi, Einsi)
Re-alignment
Tree calculation
• FastTree or IQ-TREE
• Resolution of polytomies
• calculate TBL, mean BL and median BL
Patristic distance calculation

Fig. 7: Pipeline to calculate patristic distances in a data set

The measure of phylogenetic diversity is the sum of all patristic tree-distances or the total branch length of a tree (Faith, 1992). Sherlock thus calculates for each inferred tree the total branch length, mean branch length and median branch length (Fig. 7).

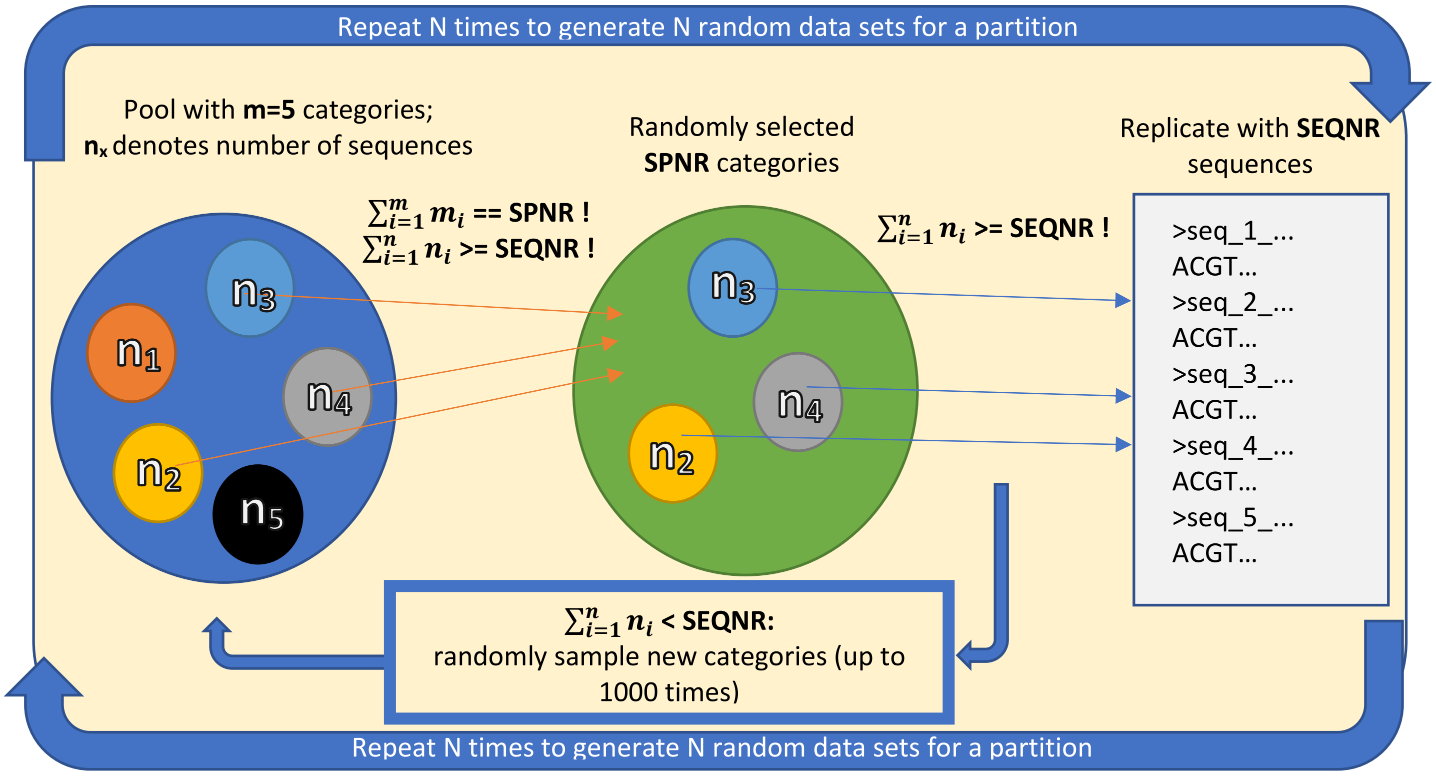
TBL, mean BL and median BL are calculated for each inferred tree from specified original sub-pool data sets. The observed patristic distances from original data are reference to the results calculated by the same pipeline for all replicates of a random sub-pool data set. The calculated patristic distance measures for all randomly sampled replicates of a sub-pool data set form a random distribution of patristic distances which is compared to the original reference value of a sub-pool.

5.4. Process of randomisation

Before the random replicates for a specified sub-pool are sampled, Sherlock samples a base pool containing all sequences from coherent sub-pools. Sherlock thus checks the combined sub-pool data for duplicates, i.e. if a sequence occurs more than once in the pool of sub-pools.

Before Sherlock starts, the user is asked whether they want to proceed with duplicates. If so, Sherlock creates a list of redundant sequence names and in which files the duplicates occurred. Even if the user wishes to continue, only one occurrence of each duplicate is kept in the pool. Hence only unique sequence names can be randomly sampled from the pool.

The random sub-pool data sets or partitions are generated by randomly sampling sequences from the pool. A pool consists of all sequences of the sub-pools it contains – that is by definition all subdirectories in a pool directory (see 3.2. Folder structure).



Repeat N times to generate N random data sets for a partition
Pool with **m=5** categories;
n$_x$ denotes number of sequences
$\sum_{i=1}^{m} m_i$ == SPNR !
$\sum_{i=1}^{n} n_i$ >= SEQNR !
n_1
n_2
n_3
n_4
n_5
Randomly selected
SPNR categories
n_2
n_3
n_4
$\sum_{i=1}^{n} n_i$ >= SEQNR !
Replicate with **SEQNR** sequences
>seq_1_...
ACGT...
>seq_2_...
ACGT...
>seq_3_...
ACGT...
>seq_4_...
ACGT...
>seq_5_...
ACGT...
$\sum_{i=1}^{n} n_i$ < SEQNR:
randomly sample new categories (up to 1000 times)
Repeat N times to generate N random data sets for a partition

Fig. 8: Randomisation algorithm for a partition: Sherlock creates REPNR replicates with SPNR categories and SEQNR sequences. First, from the partition inherited sequences, the sequences are grouped into categories. The pool must contain enough categories and sequences to be further processed. Sherlock then randomly selects categories which must contain altogether at least SEQNR sequences. If the number of sequences is insufficient, the categories have to be re-sampled. From a sufficient sequence pool, SEQNR sequences are randomly assembled into a random set of sequences to form a replicate data set.

The number of random replicates for each sub-pool or partition is given as REPNR (see 3.3.2. REPNR) in the parameter file as well as the number of categories (e.g. species, genes, locations) and the number of sequences which have to sampled for each random replicate. At best, the number of sequences and categories found in the original partitions matches the numbers in the parameter file to create a comparable distribution of random patristic distances for each original partition.

After the pool for coherent sub-pools or partitions is assembled, Sherlock randomly selects the specified number of categories (Fig. 8, see 3.3.3. SPNR), which depends on the passed naming convention keyword positions (see 3.1. Fasta file naming convention). Next, at least one sequence of each randomly selected category is added to the current random replicate data set. By that, Sherlock ensures a consistent number of categories between random replicates and the corresponding original partition data set. Afterwards, Sherlock randomly selects the remaining number of sequences necessary to equal SEQNR (see 3.3.4. SEQNR).

If a pool does not inherit enough different categories from its' partitions to meet the parameter requirements, the partition is skipped. This can happen, if not enough different categories arise from the choice of naming convention. If at least one possible combination of categories fulfils SEQNR, Sherlock proceeds with a partition.

If a pool contains enough categories and sequences to assemble matching random replicates, but a set of categories is sampled that does not consist of a fitting number of sequences, Sherlock discards it and re-samples up to 1000 times if no matching set of categories is sampled (Fig. 8). This can happen, if some categories highly outnumber the other categories in a pool.

If Sherlock cannot randomly assemble a fitting set of categories after 1000 tries, it selects the category with fewest sequences and calculates the difference to the necessary number of sequences. That category is removed and its' size added to the difference. Sherlock then searches for all remaining categories containing at least a number of sequences equal to the difference and randomly selects one of the categories.

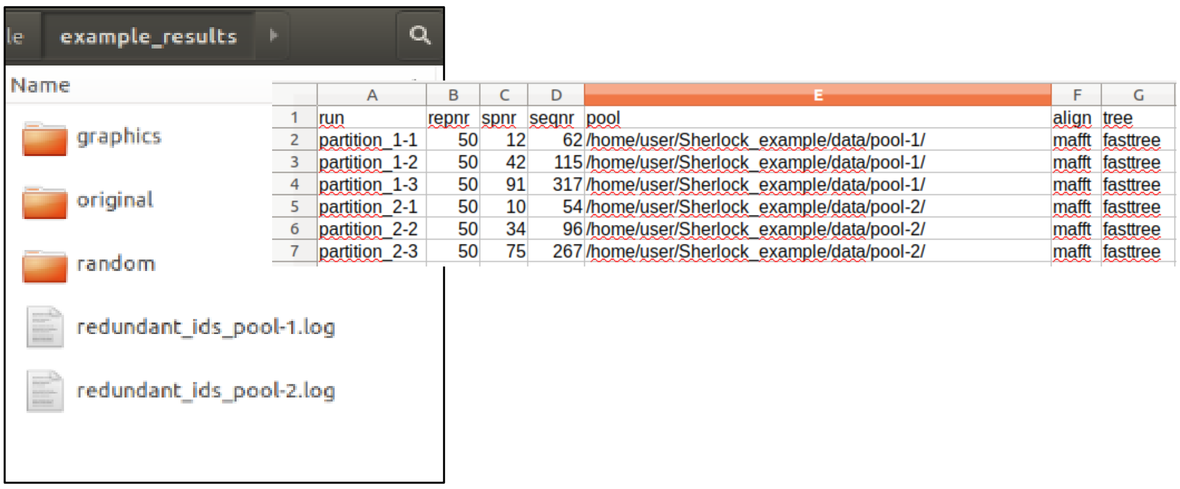
If no single category can sufficiently complete the set of categories to meet SEQNR, the second in sequences lowest category is removed, its' size added to the difference and two categories with at least the missing difference in sequences are randomly selected. Sherlock proceeds until a fitting set of categories matching SEQNR is formed.

When all random replicates for all partitions are assembled, the Sherlock pipeline is started and begins to align all random sequence files generated in the process of randomisation. In the next step, the specified tree method infers the trees for all alignments of random data and calculates the patristic distance measures (see 5.3. Pipeline to calculate patristic distances).

The results of the pipeline processes are forwarded to the R scripts plotting the random distributions.

5.5. Output and results

Sherlock writes output to the user defined output folder (see 4.1. Start Sherlock via command line) – in case the folder does not exist at program start, Sherlock creates it. After the pipeline finished, the specified output folder contains folders including the original data sets in `original/`, the random data sets in `random/` and the graphical results in `graphics/`, among other result files (Fig. 9).



example_results

Name
- graphics
- original
- random
- redundant_ids_pool-1.log
- redundant_ids_pool-2.log

	A	B	C	D	E	F	G
1	run	repnr	spnr	seqnr	pool	align	tree
2	partition_1-1	50	12	62	/home/user/Sherlock_example/data/pool-1/	mafft	fasttree
3	partition_1-2	50	42	115	/home/user/Sherlock_example/data/pool-1/	mafft	fasttree
4	partition_1-3	50	91	317	/home/user/Sherlock_example/data/pool-1/	mafft	fasttree
5	partition_2-1	50	10	54	/home/user/Sherlock_example/data/pool-2/	mafft	fasttree
6	partition_2-2	50	34	96	/home/user/Sherlock_example/data/pool-2/	mafft	fasttree
7	partition_2-3	50	75	267	/home/user/Sherlock_example/data/pool-2/	mafft	fasttree

Fig. 9: Contents written to a user specified output directory (left), based on an exemplary parameter file (right).

5.5.1. Histograms and distribution

The main results are the graphical plots of random data set distributions for each analysed partition. The histograms and density of the single partitions are saved as PDFs in `graphics/histograms+densities/` and show the random distributions for the three patristic distance measure.

The random patristic distance measure scores or branch lengths are plotted on the x-axis and the density of the distribution corresponding to the frequency on the y-axis. Within the plots (Fig. 10), the red bar marks the original data sets observed patristic distance score and the violet bar the expected score from the random data distribution. The dotted lines indicate the left and right border of the $Q_{0.95}$ quantile of the distribution.

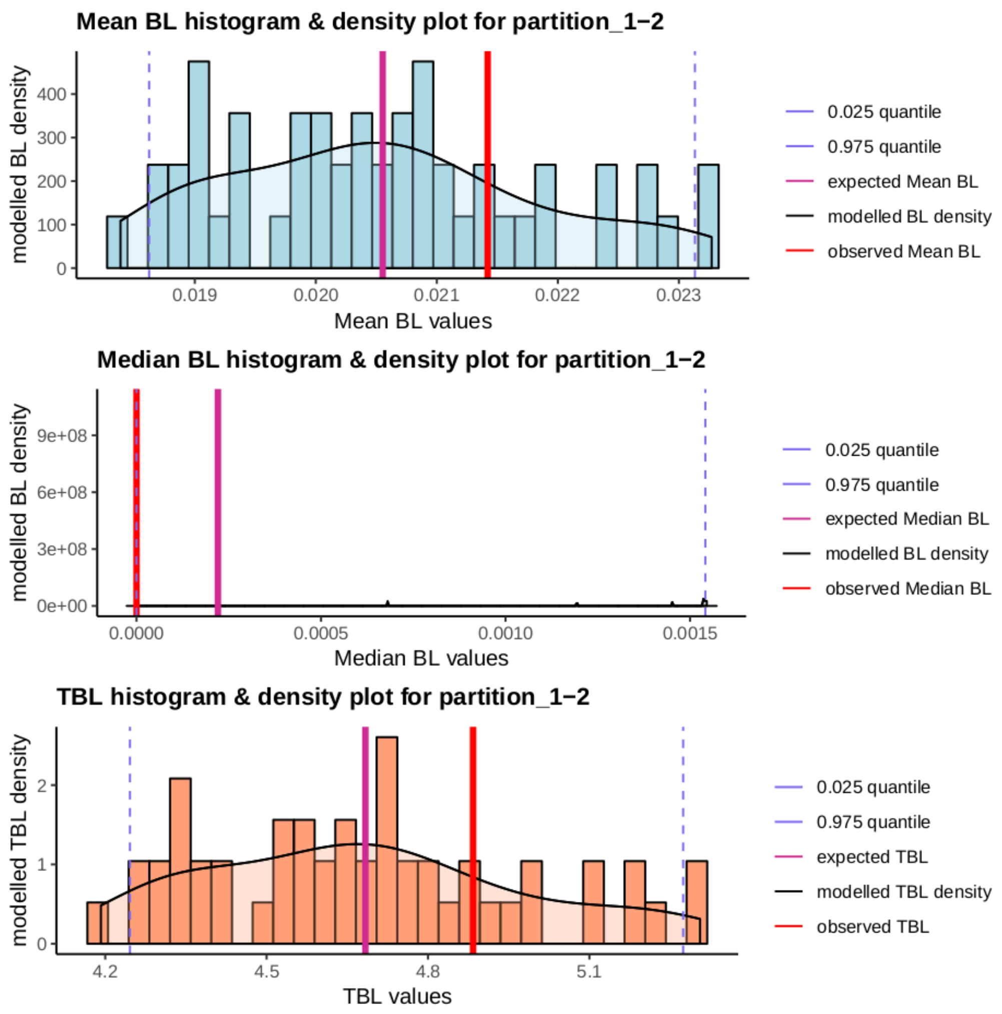
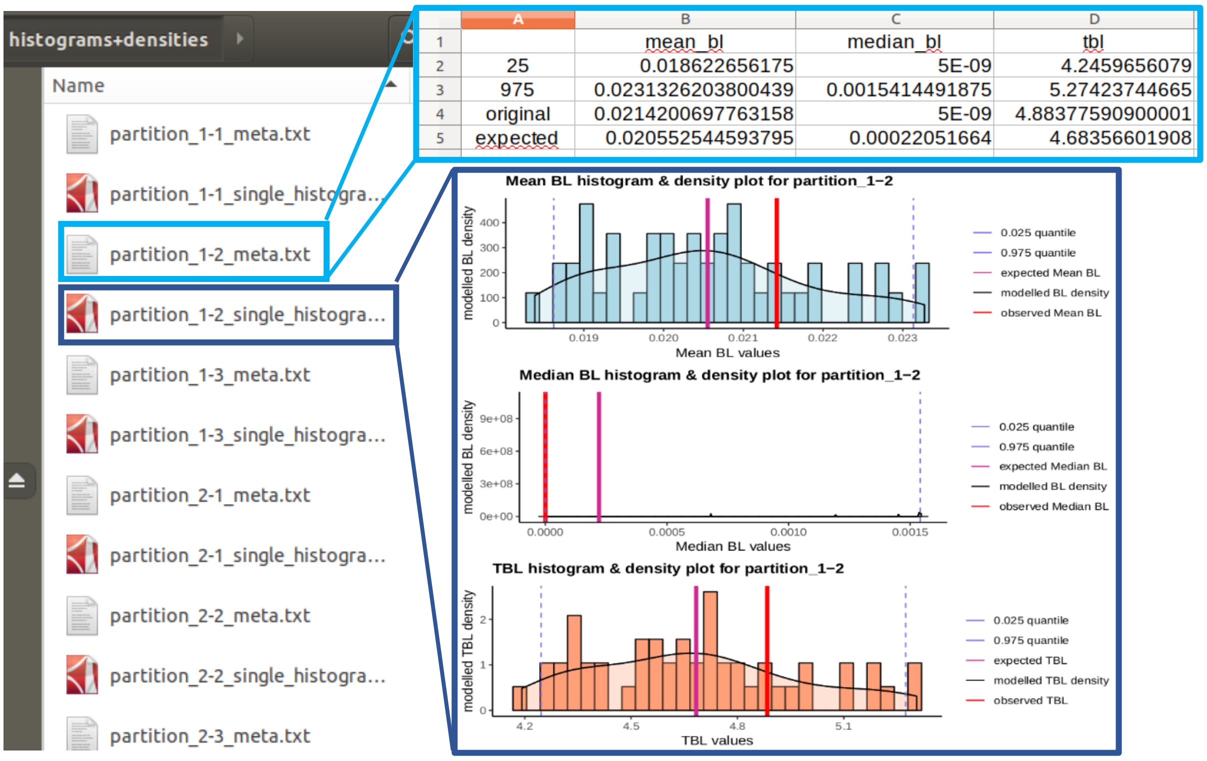


Fig. 10: Exemplary histogram plot produced by Sherlock. For each partition, the patristic distance measures (mean branch length, median branch length and total branch length) are plotted. Patristic distance scores are plotted on the x-axis, the density corresponding to the frequency of the random distribution to the y-axis. The red vertical bar indicates the original patristic distance, the violet bar the expected patristic distance score from the random distribution. The dotted lines limit the $Q_{0.95}$ *quantile. In the above graphic, none of the measures is regarded as off range, i.e. an original lying outside* $Q_{0.95}$*. Many polytomies in inferred trees result in an accumulation of branch lengths = 0 and thus the median happens to be zero, too.*

If the observed patristic distance of a partition lies outside the $Q_{0.95}$, the name of the run or partition is listed in a file called Off_range_list.txt that saved all partitions incorporating such outliers in one of the three patristic distance measure histograms. To investigate the exact values of each partition in depth, `histograms+densities/` includes a .txt file for each single histogram PDF showing the exact values for $Q_{0.025}$, $Q_{0.975}$, observed patristic distance from the original data set and expected patristic distance score from the random distribution (Fig. 11).



histograms+densities

Name

partition_1-1_meta.txt
partition_1-1_single_histogra…
partition_1-2_meta.txt
partition_1-2_single_histogra…
partition_1-3_meta.txt
partition_1-3_single_histogra…
partition_2-1_meta.txt
partition_2-1_single_histogra…
partition_2-2_meta.txt
partition_2-2_single_histogra…
partition_2-3_meta.txt

	A	B	C	D
1		mean_bl	median_bl	tbl
2	25	0.018622656175	5E-09	4.2459656079
3	975	0.0231326203800439	0.0015414491875	5.27423744665
4	original	0.0214200697763158	5E-09	4.88377590900001
5	expected	0.020552544593795	0.00022051664	4.68356601908

Fig. 11: Contents of histograms+densities/. For each created single partition or sub-pool histogram plot, a .txt file is created showing the exact values for each patristic distance measure.

The single partition histogram PDFs are merged into a single PDF located at `graphics/` as well.

5.5.2. Violin plots

Sherlock plots the patristic distance measures as violin plots, too. The violin plots are located in `graphics/violins/` and Sherlock yields six PDFs - for each of the three patristic distance measures (see 5.3. Pipeline to calculate patristic distances) two files: one file grouping the patristic distance scores according to SPNR (see 3.3.3. SPNR) and one file grouping the scores according to SEQNR (see 3.3.4. SEQNR).

The bin sizes (SPNR or SEQNR respectively) are arranged on the x-axis and the patristic distance scores on the y-axis. The violins are displayed as dot plots showing all random patristic distance score as single dot as well as boxplots (Fig. 12).

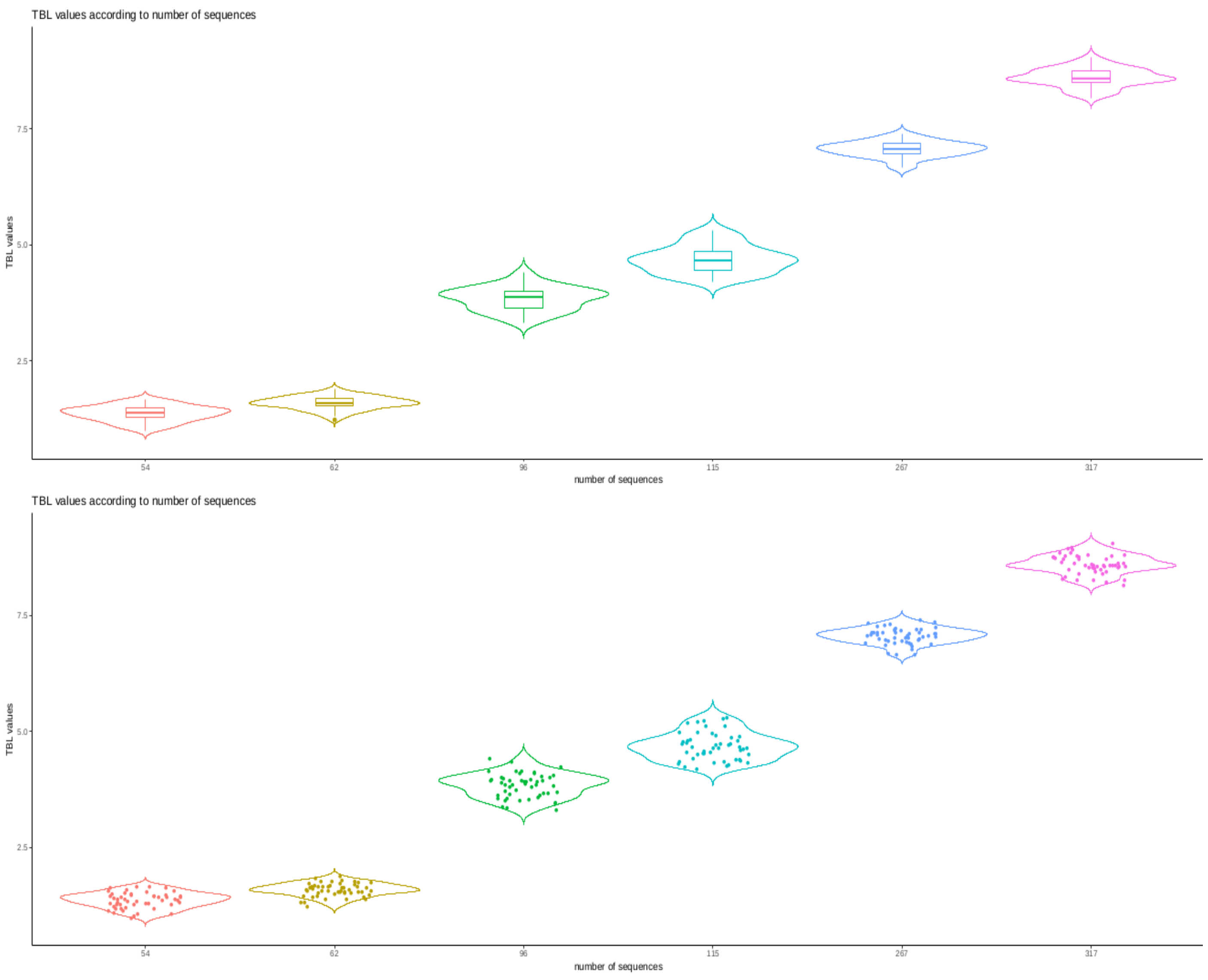


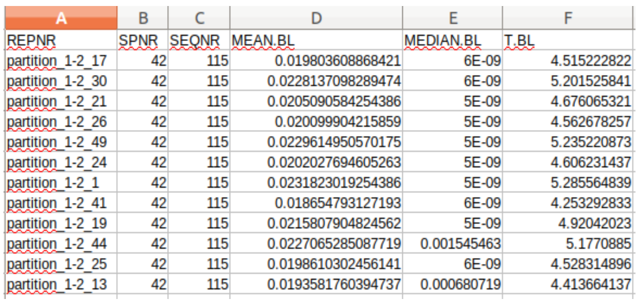
Fig. 12: Violin plot of total branch lengths calculated from random data sets and grouped after number of number of sequences (SEQNR) for each random data set. The x-axis shows SEQNR and the y-axis the patristic distance score. The above plot depicts the patristic distances as boxplots and the lower plot depicts the same sata as dotplots.

The violin plots help understand the influence of SEQNR and SPNR sizes on the patristic distance scores.

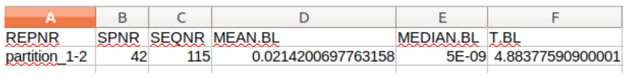
5.5.3. Other information and results

The basis of the graphical plots are the results from the calculation of the pipeline which include aligning the original and random data sets as well as the inference of trees. The original and random data sets are to be found in original/ and random/ respectively and incorporate all analysed pools and the corresponding partitions or sub-pools defined within the parameter file.

The partition folders contain the alignments of the data sets and the inferred trees. Additionally, all calculated patristic tree-distance measures for the data sets of a partition are written to a .tsv file. Since the original, non-randomised data consists of only one set of sequences, only one alignment and one tree are stored for each original partition. Contrary to that, each partition within random/ contains the by REPNR (see 3.3.2. REPNR) specified number of random replicate alignments, trees and patristic distance scores. The .tsv file consequentially stores all patristic distance measure scores for each computed random replicate.



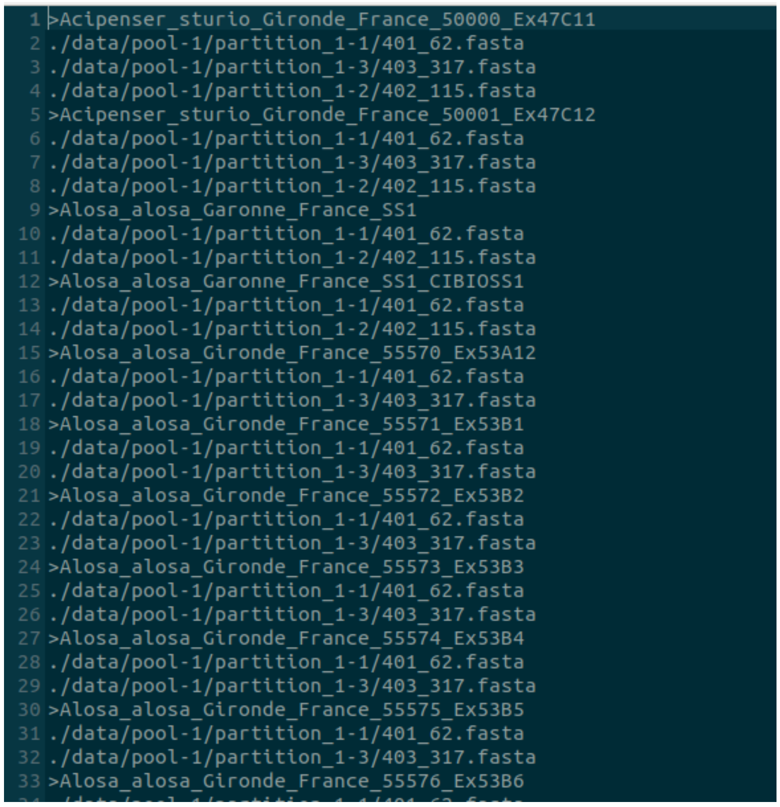
A	B	C	D	E	F
REPNR	SPNR	SEQNR	MEAN.BL	MEDIAN.BL	T.BL
partition_1-2_17	42	115	0.019803608868421	6E-09	4.515222822
partition_1-2_30	42	115	0.0228137098289474	6E-09	5.201525841
partition_1-2_21	42	115	0.0205090584254386	5E-09	4.676065321
partition_1-2_26	42	115	0.020099904215859	5E-09	4.562678257
partition_1-2_49	42	115	0.0229614950570175	5E-09	5.235220873
partition_1-2_24	42	115	0.0202027694605263	5E-09	4.606231437
partition_1-2_1	42	115	0.0231823019254386	5E-09	5.285564839
partition_1-2_41	42	115	0.018654793127193	6E-09	4.253292833
partition_1-2_19	42	115	0.0215807904824562	5E-09	4.92042023
partition_1-2_44	42	115	0.0227065285087719	0.001545463	5.1770885
partition_1-2_25	42	115	0.0198610302456141	6E-09	4.528314896
partition_1-2_13	42	115	0.0193581760394737	0.000680719	4.413664137



A	B	C	D	E	F
REPNR	SPNR	SEQNR	MEAN.BL	MEDIAN.BL	T.BL
partition_1-2	42	115	0.0214200697763158	5E-09	4.88377590900001

Fig. 13: Example of a patristic distance measure file for a single partition. The left excerpt shows the measures for all randomly sampled replicates (according to REPNR). The above image shows the measures for the original data set.

The output folder contains a redundant sequence name file (.log) and lists for every analysed pool the names of duplicate sequences and in which files the duplicates occurred.



```
>Acipenser_sturio_Gironde_France_50000_Ex47C11
./data/pool-1/partition_1-1/401_62.fasta
./data/pool-1/partition_1-3/403_317.fasta
./data/pool-1/partition_1-2/402_115.fasta
>Acipenser_sturio_Gironde_France_50001_Ex47C12
./data/pool-1/partition_1-1/401_62.fasta
./data/pool-1/partition_1-3/403_317.fasta
./data/pool-1/partition_1-2/402_115.fasta
>Alosa_alosa_Garonne_France_SS1
./data/pool-1/partition_1-1/401_62.fasta
./data/pool-1/partition_1-2/402_115.fasta
>Alosa_alosa_Garonne_France_SS1_CIBIOSS1
./data/pool-1/partition_1-1/401_62.fasta
./data/pool-1/partition_1-2/402_115.fasta
>Alosa_alosa_Gironde_France_55570_Ex53A12
./data/pool-1/partition_1-1/401_62.fasta
./data/pool-1/partition_1-3/403_317.fasta
>Alosa_alosa_Gironde_France_55571_Ex53B1
./data/pool-1/partition_1-1/401_62.fasta
./data/pool-1/partition_1-3/403_317.fasta
>Alosa_alosa_Gironde_France_55572_Ex53B2
./data/pool-1/partition_1-1/401_62.fasta
./data/pool-1/partition_1-3/403_317.fasta
>Alosa_alosa_Gironde_France_55573_Ex53B3
./data/pool-1/partition_1-1/401_62.fasta
./data/pool-1/partition_1-3/403_317.fasta
>Alosa_alosa_Gironde_France_55574_Ex53B4
./data/pool-1/partition_1-1/401_62.fasta
./data/pool-1/partition_1-3/403_317.fasta
>Alosa_alosa_Gironde_France_55575_Ex53B5
./data/pool-1/partition_1-1/401_62.fasta
./data/pool-1/partition_1-3/403_317.fasta
>Alosa_alosa_Gironde_France_55576_Ex53B6
```

Fig. 14: Excerpt from a redundant sequence names file. First, a sequence name is listed and followed by all file names where the sequence name occurred. Such a file is created for all pools analysed in a parameter file.

Citation

To cite Sherlock in a publication, please use

Seidel, N. (2019). Sherlock: An Automated Process Pipeline for Statistical Evaluation of Patristic Tree-Distance. *Version 1.0.* Retrieved from https://www.github.org/NathanSeidel/Sherlock

A BibTeX entry for LaTeX users is

```
@Manual{,
     title = {Sherlock: An Automated Process Pipeline for
     Statistical Evaluation of Patristic Tree-Distance Variation},
     author = {Nathan Seidel},
     organization = {Zoological Research Museum Alexander Koenig},
     address = {Bonn, Germany}
     year = {2019},
     note = {version 1.0},
     url = {https://www.github.org/NathanSeidel/Sherlock},
}
```

License

If you have any problem, error-reports or other questions about Sherlock, feel free and write an email to nseidel@uni-bonn.de .

References

Auguie, B. (2017). gridExtra: Miscellaneous Functions for "Grid" Graphics. *R package version 2.3.* Retrieved from https://CRAN.R-project.org/package=gridExtra

Faith, D. P. (1992). Conservation Evaluation and Phylogenetic Diversity. *Biol. Conserv.*(61), pp. 1-10.

Katoh, K., & Standley, D. M. (2013). MAFFT multiple sequence alignment software version 7: improvements in performance and usability. *Mol Biol Evol., 30*(4), pp. 772–780. doi:10.1093/molbev/mst010

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., & Leisch, F. (2019). e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071). *R package version 1.7-1.* TU Wien. Retrieved from https://CRAN.R-project.org/package=e1071

Nguyen, L.-T., Schmidt, H. A., von Haeseler, A., & Minh, B. Q. (2015). IQ-TREE: A fast and effective stochastic algorithm for estimating maximum likelihood phylogenies. *Mol. Biol. Evol.*(32), pp. 268-274. Retrieved from https://doi.org/10.1093/molbev/msu300

Paradis, E., & Schliep, K. (2018). ape 5: an environment for modern phylogenetics and evolutionary analyses in R. *Bioinformatics*(35), pp. 526-528.

Price, M. N., Dehal, P. S., & Arkin, A. P. (2010). FastTree 2 -- Approximately Maximum-Likelihood Trees for Large Alignments. *PLoS ONE, 5*(3), p. e9490. doi:10.1371/journal.pone.0009490

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis.* New York: Springer-Verlag.