MatPhylobi ver. 0.2.0

Instruction Manual

Hanna Kranas, Krzysztof Spalik and Łukasz Banasiak

February, 2019

Table of contents

Introduction

MatPhylobi is a simple command-line program that constructs taxonomic datasets for phylogenetic inference based on NCBI (GenBank) data. Provided with taxon name or list, a set of molecular markers with exemplary reference nucleotide sequences, and, optionally, a dictionary of synonyms and lists of excluded taxa and excluded GenBank accessions, MatPhylobi returns FASTA files for each marker with each species represented by a single GenBank accession. Additionally, for each species and each marker, it reports the number of sequences deposited in GenBank from which it has picked up the best representative and the GenBank accession version identifier of the latter. Particular functionality of MatPhylobi is its ability to concatenate nrDNA ITS1 and ITS2 sequences from the same specimen that were deposited in GenBank separately.

The main applications of MatPhylobi are phylogenetic inference and screening molecular data coverage for a specific taxonomic group. This tool was particularly designed to assist in large phylogenetic and taxonomic studies. Usually, the data downloaded from GenBank require a lot of work prior to the phylogenetic analyses. Despite quality control in GenBank, in large datasets there are always accessions that are misidentified, presumed chimeric or containing apparent sequencing errors that may interfere with phylogenetic inference; these accessions have to be excluded from the analyses. Additionally, some specific names need to be updated following recent taxonomic revisions that are not always reflected in GenBank taxonomic system yet. Sometimes, conspecific accessions are deposited in GenBank under different synonymous names. And, last but not least, each species has to be represented by a single accession for each marker. This time-consuming procedure has to be repeated with each update of the dataset obtained from GenBank. MatPhylobi facilitates such updates because it can reuse the lists of excluded taxa and accessions and the dictionary of synonymous names. When choosing the representative sequence for each species and marker, the algorithm determines which sequence aligns best to all sequences for a given species. This approach helps to avoid the choice of outliers when more than two sequences are found in GenBank.

Requirements

MatPhylobi is distributed without charge and can be downloaded from https://github.com/hansiu/MatPhylobi

The program is written in Python and has the following dependencies:

- Python > 3.4
- Biopython > 1.68
- BLAST+ > 2.5.0
- optional: BLAST databases, see section “BLAST databases”

Internet connection is required for running MatPhylobi, because it uses Entrez tools to communicate with GenBank even if BLAST databases are installed on a local machine. MatPhylobi can be forced to run using earlier versions of Python than 3.4 but it is NOT recommended as some advanced functionalities will terminate with an error. The program can be run on Linux, Mac OS and MS Windows.

Package contents

MatPhylobi package contains the following files:

- program files in ‘MatPhylobi’ subfolder
- instruction manual (this document)

- a folder containing examples of input files:
    - orgs.txt – a list of included taxa
    - excl.txt – a list of excluded taxa
    - synonyms.txt – a dictionary of taxonomic synonyms
    - blacklist.txt – a list of excluded GenBank accessions
    - config.ini – a configuration file

Installation of MatPhylobi

MatPhylobi was developed and tested in Linux environment. However, it may be run under Windows and Mac OS if all the dependencies are properly installed. MatPhylobi itself is a Python package to be installed using a standard `setup.py` script.

After installation, MatPhylobi is available by typing `MatPhylobi` in the console. For usage info please type:

`MatPhylobi analyze -h` or `MatPhylobi inputfile -h` or `MatPhylobi update -h`

Linux

The following detailed installation instructions concern Ubuntu – popular Linux distribution – and require typing commands in the terminal (console). You must have administrative privileges to your system. Before starting installation of MatPhylobi, make sure that your system is up to date by running the following commands:

```
sudo apt update
```

```
sudo apt upgrade
```

Ubuntu 16.04 LTS

In this distribution of Ubuntu, Python 3.5.2 is already preinstalled and can be launched using `python3` command. The easiest way to install current version of Biopython is to install the PyPA recommended tool (pip) for installing Python packages first (it may take several minutes on fresh systems):

```
sudo apt install python3-pip
```

and then subsequently Biopython using pip (this step will also automatically install Python package 'numpy' required by Biopython):

```
sudo pip3 install biopython
```

Next, to download BLAST+, check to find the newest version and change the following line accordingly:

```
wget ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/ncbi-blast-2.8.1+-x64-linux.tar.gz
```

When the proper package has been downloaded, the installation process is to open the archive and copy the binaries into a directory included in the system PATH variable, for instance `/usr/local/bin`. You can check directories included in the PATH variable by typing `echo $PATH`.

```
tar -zxf ncbi-blast-2.8.1+-x64-linux.tar.gz
```

```
sudo cp ./ncbi-blast-2.8.1+/bin/* /usr/local/bin
```

Next step is to unpack MatPhylobi archive:

```
unzip MatPhylobi-master.zip
```

You can alternatively clone GitHub repository:

```
git clone https://github.com/hansiu/MatPhylobi.git MatPhylobi-master
```

Then, enter the directory and run the installation script:

```
cd MatPhylobi-master
```

```
sudo python3 setup.py install
```

Ubuntu 17.10 and newer

On relatively new Ubuntu distributions MatPhylobi's installation is fairly easy due to the updates in the repositories. However, setup tools for Python are no longer included within Python distribution and must be installed individually:

```
sudo apt install python3-setuptools
```

Biopython can be installed directly from Ubuntu repository (there is no need for installing pip first):

```
sudo apt install python3-biopython
```

BLAST+ is installed automatically during the Biopython installation. All remaining steps are the same as for Ubuntu 16.04 LTS – the last two steps are to unpack MatPhylobi's archive and to run the installation script.

Unpack the archive:

```
unzip MatPhylobi-master.zip
```

The alternative is to clone GitHub repository:

```
git clone https://github.com/hansiu/MatPhylobi.git MatPhylobi-master
```

Finally, enter the directory and run the installation script:

```
cd MatPhylobi-master
```

```
sudo python3 setup.py install
```

MS Windows

MatPhylobi was tested to run on Windows 8 and Windows 10.

Most Windows versions do not have Python pre-installed but user friendly web-based or executable installer can be downloaded from https://www.python.org/downloads/windows/ At the first screen displayed, please ensure that 'Add Python 3.4 to PATH' check box is selected.

As the default installation of Python includes pip – the PyPA recommended tool for installing Python packages – installation of Biopython is fairly easy. Open the command line window (i.e., launch `cmd.exe`) and type:

```
pip install biopython
```

Standalone BLAST setup for MS Windows including setting up the optional BLAST databases is described in the following document https://www.ncbi.nlm.nih.gov/books/NBK52637/

Then, unpack the MatPhylobi's zip archive and open command line window once again. Navigate to the directory containing program files (the following example assumes that MatPhylobi was unpacked in 'Downloads' folder of user 'kowalski'):

```
cd C:\Users\kowalski\Downloads\MatPhylobi-master
```

and run the installation script:

```
python setup.py install
```

Finally, restart the computer. This is required for MatPhylobi to see fresh installation of BLAST+.

Mac OS X

All recent Mac OS versions have preinstalled Python 2 which can be run in terminal by typing `python` command. However, MatPhylobi requires Python 3.4 or newer and this must be installed additionally. Please download user friendly installer from https://www.python.org/downloads/mac-osx/ and follow the instructions on the screen. After installation, Python 3 is available through `python3` command.

Next, install pip – the PyPA recommended tool for installing Python packages. Open the terminal and type:

```
sudo easy_install pip
```

Then, installation of Biopython is fairly easy by typing:

```
pip3 install Biopython
```

Ensure that the number '3' in `pip3` command has not been omitted. Otherwise, Biopython will be installed for Python 2 instead of Python 3.

Standalone BLAST for Mac can be installed using a standard installer downloaded from ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/ For instance, BLAST+ ver. 2.8.1 has the installer named `ncbi-blast-2.8.1+.dmg`.

Having all the dependencies installed, unpack MatPhylobi package by double clicking the icon and then open the terminal and navigate to the folder containing program files. Finally run the installation script by typing:

```
python3 setup.py install
```

BLAST databases

Installing BLAST databases is optional and generally unnecessary for small projects involving several dozen taxa and a few molecular markers. For large analyses, however, it is better to have BLAST databases downloaded locally because it speeds up the calculations and makes the program execution less dependent on the speed and reliability of network connection. If you decide to download and install BLAST databases, you must remember to update them regularly as **outdated database may lead to a crash** of the analysis or misleading results. The following installation instructions concern Ubuntu – popular Linux distribution. For MS Windows, please refer to the following document https://www.ncbi.nlm.nih.gov/books/NBK52637/

Installation

First, install additional tools for Perl from Ubuntu repository:

```
sudo apt install liblist-moreutils-perl
```

Next, make a directory for the files storing the data – these are quite large (about 55 GB in May 2018 and still growing) so it is not recommended to use solid-state storage devices (SSD) with limited capacity – and let your system know where you intended to store the database by exporting a BLASTDB variable:

```
mkdir MatPhylobiBlast
```

```
export BLASTDB="/home/kowalski/MatPhylobiBlast"
```

As the variable must be declared before each usage of BLAST databases, the most convenient is to include `export ...` command in a file executed on a system start up, for instance `.profile` on Linux. You can always check if BLASTDB variable has been declared properly by typing `echo $BLASTDB`.

Finally, navigate to the directory and run Perl script being a part of BLAST+ package:

```
cd MatPhylobiBlast
```

```
update_blastdb.pl --passive --decompress nt taxdb
```

Automatic updates

In Linux you can use `crontab` to schedule periodical runs of the above script. Type `contab -e` to enter the scheduled tasks editor and then add a new line according to the displayed documentation and your needs. It is recommended to check for updates every day when the database usage is minimal, usually late in the night:

```
0 3 * * * update_blastdb.pl --passive
```

The above line sets up the update script to be run every day at 3 AM. The script will not download any data if all files are up to date.

Getting Started

Once MatPhylobi is installed in the system, it can be run by typing in the terminal a command matching the following pattern:

```
MatPhylobi <mode_command> <arguments> [--api_key <APIKEY>]
```

Providing a valid API key after optional `--api_key` argument can speed up the analysis from three to ten requests per second. The API key identifies an NCBI account and is passed to Biopython Entrez by MatPhylobi.

Mode commands

MatPhylobi can be run in two basic modes: `analyze` and `update`. The `analyze` command sends a new query to GenBank. The `update` command repeats exactly the same query but searching only for sequences submitted to GenBank since the previous query. Then the whole analysis is repeated only for those taxa and molecular markers for which new data are available. The resulting FASTA files are updated and changes to the GenBank data are reported.

Analyze mode

The `analyze` command requires at least three obligatory arguments, `-g`, `-s`, and `-o`, specifying gene name, GenBank ID of its example sequence, and taxon name, respectively:

```
MatPhylobi analyze -g <gene_name> -s <GenBankID> -o <taxon_name>
```

Gene name argument is `-g` or `--gene_names`.

At least one name has to be specified. If more names are provided, they should be separated by spaces. There are no restrictions for the names, except for ITS, ITS1, and ITS2, which are always interpreted as the nuclear ribosomal DNA internal transcribed spacers. Examples:

```
-g ITS
--gene_names ITS
-g ITS rps16 rpoC1
```

Sequence argument is `-s` or `--sequences`.

When two or more markers are specified, reference accessions must be provided in the same order as gene names, otherwise MatPhylobi will return misleading results. Use comma to separate reference accessions for the same marker and a space to separate different markers. All sequences for a given molecular marker should be obtained from the same strand. If a particular sequence comes from a complementary strand, then a letter 'R' must be added at the end of its accession number, e.g. 'KJ832104.1R' instead of 'KJ832104.1'. Examples:

```
-s FJ415117.1
--sequences FJ415117.1
-s FJ415117.1,AF077779.1,KF806584.1 KJ832104.1,GU395133.1
```

Organism argument is `-o` or `--org_included`.

At least one taxon has to be included. Names must be provided according to the GenBank taxonomy. If multiple names are provided, they should be separated by spaces. If the name of a group consists of more than one word, spaces between the words must be replaced by a plus '+' sign, otherwise will be recognized as separate names. Examples:

```
-o Daucinae
--org_included Daucinae
-o Laserpitium Anthriscus Daucus+carota
```

Included taxa may also be listed in a separate file. The path and the filename should be specified using `--org_included_file </path/filename>`.

An example of a simple query is below. This query will download nrDNA ITS sequences of *Laserpitium*, one accession for each species, using ITS sequence of *Laserpitium latifolium*, the generitype of *Laserpitium*, as an example.

```
MatPhylobi analyze -g ITS -s FJ415131.1 -o Laserpitium
```

Inputfile mode

Alternatively, all the arguments for analyse mode may be given in a configuration file. Use the `inputfile` mode command followed by `--file` argument with the path and filename:

```
MatPhylobi inputfile --path </path/filename> [-f </path>]
```

Optionally, the output folder may be specified using `-f` argument. All other arguments are ignored when using `inputfile` mode.

Update mode

Update run requires only the `update` command after `MatPhylobi` call and an access to the files from the previous run that are to be updated. The files are assumed to be in the current directory; alternatively, the folder may be indicated using `-f` argument. Any other arguments, are ignored as the settings are taken from the `config.ini` configuration file created during the previous run including the dictionary of synonyms and the blacklist of accessions. **Editing or deleting any of these three files since the previous run can lead to unexpected results** as they would have effect only for taxa and molecular markers for which new sequences are available in GenBank, and thus are reanalysed. Other parts of the dataset would remain untouched and applying different setting during update mode could introduce inconsistencies among new and old data. Example:

```
MatPhylobi update [-f </path>]
```

If there is a need to reanalyse the whole dataset using updated dictionary of synonyms and blacklist of accessions, the `analyze` mode should be used instead, utilizing previously generated `config.ini` file. Contrary to the `update` mode, the configuration file can be safely edited for the `inputfile` mode to change required settings. For an example see 'Inputfile mode' section.

Input files

Names of the input files containing lists of included and excluded taxa, list of synonym, and blacklisted sequence identifiers are free to choose and can be placed in any directory – their path can be specified using corresponding arguments on the command line.

List of included taxa

Taxa included in the analyses may be specified either using the respective command or listed in a file. In the latter case, the file should include at least one taxon name. If more taxa are provided, each should be given in a new line. Names must follow the GenBank taxonomy. Please note that this system is unranked and contains both formal names of taxa and informal names of clades.

List of excluded taxa

Sometimes, we need to exclude particular taxa from the analyses. For example, these are species that are known to introduce topological conflicts because they are of hybrid origin or their sequences are highly evolved and subject to long-branch attraction. These taxa may be specified either using the respective command or listed in a file. The format of the file is the same as for included taxa.

Dictionary of synonyms

If two or more species names are considered synonyms but they are given separate taxonomy identifiers (txid) in GenBank Taxonomy, a list of synonymous IDs can be provided by the user. MatPhylobi uses this list at the beginning of the analysis prior to the selection of representative sequences. Accessions of taxa indicated as synonymous are therefore considered as conspecific in subsequent steps. This dictionary is a text file containing comma-separated lists of taxonomy identifiers preceded by a proper taxon name, each species in a new line:

```
<taxon name>, <txid_to_join1>, <txid_to_join2>, ...
```

For example, *Laserpitium petrophilum* and *Ekimia petrophila* have different taxonomy IDs in the GenBank: taxid109128 and taxid1143247, respectively. However, these names are nomenclatural (homotypic) synonyms with *Ekimia* being the correct generic placement. The line in the dictionary should be as follows:

```
Ekimia petrophila, 1143247, 109128
```

Alternatively, the user may use the dictionary just for renaming a taxon represented in GenBank taxonomy as a single txid.

For example, *Conopodium capnoides* (txid109132) is not related to its presumed congeners but to the species of *Kozlovia* and was transferred to the latter, which decision has not been yet recognized in GenBank taxonomy. The respective entry in the dictionary should be:

```
Kozlovia capnoides, 109132
```

Comments can be added at the end of each line after a hash '#' sign.

Blacklist of sequences

A list of accessions excluded from the analysis, for instance those obviously misidentified, presumed chimeric, or containing apparent sequencing errors, may be specified by the user. The format is a simple text file containing one accession version or GI identifier per line. Comments can be added to each line after a hash '#' sign.

Configuration file

During each run MatPhylobi generates a configuration file that reflects command line arguments provided by the user. In order to repeat the analysis, instead of typing arguments in the command line, a configuration file can be used as a program input. A configuration file can be edited in any text editor and the format is the following (the values meant to be edited by a user are highlighed in gray):

```
[StartValues]
included_organisms = ['Taxon1','Taxon2', ...]
excluded_organisms = ['Taxon3', ...] or None
gene_names = ['marker1', 'marker2', ...]
sequences = [['Accession1.2','Accession2.1'], ['Accession3.1'], ...]
length_threshold = min:max
its_joiner = number_from_range1-3 or None
query_cover = [number1, number2] or None
identity = [number1, number2] or None
string_distance = number
subspecies = True or False
remote_blast = True or False
blacklist = path_to_file or empty
synonyms = path_to_file or empty
```

IMPORTANT! Relative paths, e.g. `./MatPhylobi/example_input/blacklist.txt` are interpreted in the context of current directory in which MatPhylobi is being executed and not in the context of configuration file path. To avoid confusion it is recommended to use whenever possible absolute paths, e.g. `/home/kowalski/example.txt` instead of relative paths.

Examples

Simple analyze mode

```
MatPhylobi analyze -g ITS -o Laserpitium -s FJ415131.1 -f ./Laser -b
```

MatPhylobi will search for ITS sequences of species of *Laserpitium* using a reference sequence deposited in GenBank under FJ415131.1 accession version identifier. Results will be saved in the `Laser` subdirectory of the current folder. MatPhylobi will use BLAST databases accessed online as specified by `-b` argument. By default, the sequences of infraspecific taxa will be pooled with those identified only to the species level. ITS1 and ITS2 sequences that were deposited separately in GenBank will not be concatenated.

Inputfile mode

```
MatPhylobi inputfile --path ./Laser/config.ini -f ./Laser2
```

MatPhylobi will search for sequences of molecular markers based on the arguments listed in configuration file found in path `./Laser/config.ini` and the results will be saved to `Laser2` directory. Such a run is particularly useful if one wants to repeat an analysis based on configuration file automatically created in one of the previous runs. Such a file can be edited in order to modify the original analysis.

Update mode

```
MatPhylobi update -f ./Laser
```

MatPhylobi will update the results of a previous run stored in `./Laser` directory. If `-f` argument is omitted, MatPhylobi will assume the current folder as a directory containing files to be updated.

Advanced analyze mode

```
MatPhylobi analyze -o Daucus -s KF806584 GU395133 -g ITS rps16 -f Daucus --subsp -i
```

MatPhylobi will search for sequences of genus *Daucus* using reference sequences KF806584 and GU395133 for ITS and *rps16* intron, respectively. Results will be saved in `Daucus` directory and sequences from infraspecific taxa will not be pooled but each taxon (e.g., subspecies, varieties, forms) will be itemized in the resulting FASTA files and reports. Argument `-i` specifies that sequences of ITS1 and ITS2 will be concatenated only if there is a strong evidence that they were obtained from the same specimen. By default, as `-b` argument has not been provided, MatPhylobi will attempt to use local BLAST databases.

```
MatPhylobi analyze -o Apiaceae -s AJ431081.1,FJ385176.1 AJ429370.1 -g rps16 matK -f ./analyses/Apiaceae -q 30 -y 40 20 -l 50:2000
```

In this example, MatPhylobi will search for the accessions representing family Apiaceae using two sequences of *rps16* intron (AJ431081.1 and FJ385176.1) and one sequence of *matK*. Results will be saved in `./analyses/Apiaceae` relative path. By default, the sequences of infraspecific taxa will be pooled with those at the species rank and ITS1 and ITS2 sequences will not be concatenated. Additionally, there are three arguments setting thresholds for the BLAST query:

- `-q` specifies the minimal query coverage for both markers (30%)
- `-y` sets the minimal identity of sequences (40% for *rps16* and 20% for *matK*)
- `-l` defines the range of sequence length (50–2000 bp)

This analysis covers the entire family Apiaceae comprising about 3,800 species, of which c. 1,800 are represented in GenBank by about 77,000 of sequences in total. Due to connection limitations and a large volume of data produced by BLAST, it may last for an hour. For such analyses, it is recommended to use tools like ‘nohup’, ‘screen’ or ‘tmux’ available for UNIX alike systems to run the command in the background.

IMPORTANT! Large analyses are prone to unreliable and/or slow network connections. It is recommended to run large queries to GenBank at night of the US local time due to less network load and thus lower probability of interrupting the analysis.

The resulting files are additive tables or sequences in FASTA format and thus large analysis can be divided in smaller parts each covering narrower taxonomic context. Those analyses must not be run in parallel but one after another as GenBank recognises IP address or API-key and blocks high frequency queries anyway.

Output files

The output files are stored in a current folder by default or in a directory specified with `-f` or `--folder` argument. Subfolder 'normal' is created to store the results of the analyze mode and subfolder 'update' is created in the update mode to store additional sequences. Detailed descriptions of output files are given below.

Configuration file (config.ini)

This file contains the arguments and parameters of the run. Additionally, it stores the date when the analysis was started or the date of its successful completion.

Sequence files

Sequences selected by MatPhylobi are downloaded in FASTA format and stored in files with '.fasta' extension, each molecular marker in a separate file with taxon and molecular marker included in its name, e.g. `ApialesITS_acc.fasta`. Each organism—species or infraspecific taxon, if argument `--subsp` or `-p` is invoked—is represented by one sequence. Joined ITS1 and ITS2 sequences are represented as a continuous contig.

Report files

For each species and molecular marker, the number of sequences available in GenBank and the GenBank accession version identifier of the chosen sequence are provided in files: `num_matrix.csv` and `accver_matrix.csv`, respectively. The files are in tabular format with table header in the first line. Fields are separated by semicolons instead of commas because the latter are used to separate ITS1 and ITS2 sequences.

num_matrix.csv

For each species or infraspecific taxon, the number of available sequences of each marker based on the BLAST query is provided. For the nrDNA ITS region, ITS1 and ITS2 sequences that were deposited separately in GenBank are counted as one accession.

accver_matix.csv

For each species and each molecular marker, the GenBank accession version identifier corresponding to the sequence saved in the FASTA file is provided. When MatPhylobi joined ITS1 and ITS2 sequences deposited in GenBank separately, their identifiers are given in brackets and separated by a comma, e.g. '(AF324367.1,AF324394.1)'. The first identifier corresponds to ITS1 and the second to ITS2. If only ITS1 or only ITS2 is present, missing spacer is identified with a dash, e.g. '(AF324354.1,–)'. If there are no sequences available for a particular taxon and the molecular marker, a respective cell of the matrix contains a dash '–'.

List of arguments

`-h`, `--help`

Help shows all MatPhylobi's arguments with information about their input format, description and default values. Examples:

```
MatPhylobi -h
MatPhylobi --help
MatPhylobi analyze -h
MatPhylobi inputfile -h
MatPhylobi update -h
```

`-o`, `--org_included`

It specifies the taxonomic scope of BLAST query. At least one taxonomic group or species must be included in the analysis. Names must be provided according to the GenBank Taxonomy. If multiple names are provided, they should be separated by spaces. If the name of a group consists of more than one word, the words must be joined by a plus ‘+’ sign, otherwise these will be recognized as two separate names.

This argument can be used together with or instead of `--org_included_file`. Examples:

```
-o Daucinae
--org_included Daucinae
-o Laserpitium Anthriscus Daucus+carota
```

`--org_included_file`

An alternative to `-o` argument, it gives a path to the file containing a list of taxonomic groups that should be included in the analysis (one per line). Names must be provided according to the GenBank Taxonomy. This argument can be used with or instead of `--org_included`. Example:

```
--org_included_file orgs.txt
```

`-g`, `--gene_names`

It specifies the name(s) of molecular marker(s). Marker names should be separated by space. There are no restrictions for the names, except for ITS, ITS1, and ITS2, which are always interpreted as nuclear ribosomal DNA Internal Transcribed Spacer sequences. Examples:

```
-g ITS
--gene_names ITS
-g ITS rps16 rpoC1
```

`-s`, `--sequences`

It provides a list of reference accessions for each molecular marker. Use comma to separate reference accessions for the same marker and a space to separate different markers. All the sequences for a given molecular marker should be obtained from the same strand. If a particular sequence comes from a complementary strand, then a letter ‘R’ must be added at the end of its accession number, e.g. ‘KJ832104.1R’ instead of ‘KJ832104.1’. Reference accessions must be provided in the same order as gene names, otherwise MatPhylobi will return misleading results. Examples:

```
-s FJ415117.1
--sequences FJ415117.1
```

```
-s FJ415117.1,AF077779.1,KF806584.1 KJ832104.1,GU395133.1
```

`-b`, `--remote_blast`

If the BLAST databases are not installed in the system, it is possible to access them online. Please note that the analysis will be much slower due to the necessity of downloading high loads of data. Examples:

```
-b
--remote_blast
```

`-f, --folder`

It specifies a path to the directory for storing the results. If it does not exist, MatPhylobi will attempt to create it. The default directory is the current folder. Examples:

```
-f phylogeny/Daucus
--folder phylogeny/Daucus
```

`--synonyms`

It specifies a path to the file containing a list of synonyms grouping GenBank txids and optionally renaming a given taxon. Example:

```
--synonyms synonyms.txt
```

`--blacklist`

It specifies a path to the file containing the list of GenBank accession number or GIs meant to be excluded from the analysis. Example:

```
--blacklist blacklist.txt
```

`-i`, `--its`

This option allows to join nrDNA ITS1 and ITS2 sequences that were deposited separately in GenBank. Two matching sequences are pasted using a repeat of 100 ‘N’, i.e. undefined nucleotides, as a separator. The depth of the algorithm depends on the number of letters ‘i’ provided. If the option `-i` or `--its` is chosen, ITS1 and ITS2 sequences will be joined only if they are referred to the same specimen voucher and were published by the same authors. Adding one more ‘i’ (option `-ii` in effect) causes an attempt to join the remaining ‘free’ ITS1 and ITS2 segments using GenBank txid and specimen voucher feature. Using `-iii` option is a ‘brute force’ that concatenates the remaining sequences solely if they come from the same taxon. Ties are breaking down by sorting the accession numbers – see details in ‘Recognizing and joining ITS1 and ITS2 sequences’ section. Moreover, argument `-d` argument has a strong effect on concatenating the spacers. Examples:

```
-i
--its
-ii
-iii
```

`-y`, `--identity`

Identity parameter describes the minimal percentage of similarity of the query sequence to the reference sequence, calculated by BLAST. Sequences below the threshold will be excluded from the analysis. A user can either provide one number for all the markers or a list of numbers, each corresponding to one marker. The order of numbers separated by spaces must follow the order of markers declared using `-g` argument. The default value of the threshold is 0 for all markers. Examples:

```
-y 30
--identity 30
-y 30 50 50
```

`-q`, `--query_cover`

Query coverage parameter describes the minimal percentage coverage of the reference sequence by the query sequence, calculated by BLAST. Sequences below the threshold will be excluded from the analysis. A user can either provide one number for all the markers or a list of numbers, each corresponding to one marker. The order of numbers separated by spaces must follow the order of markers declared using `-g` argument. The default value of the threshold is 0 for all markers. Examples:

```
-q 10
--query_cover 10
-q 0 30 20
```

`-l`, `--len_threshold`

It specifies length range of the sequences to be processed by BLAST and included in the analysis. The format is *minimum_length*:*maximum_length*. The default range is 50:5000. Examples:

```
-l 100:3000
--len_threshold 100:3000
```

`-p`, `--subsp`

By default, MatPhylobi pools all sequences of a given species in order to choose the best representative. This option allows infraspecific taxa—subspecies, varieties, forms, etc.—to be recognized as separate in resulting FASTA files and tabular reports. Hybrids, clones and specimens not identified to species level are always ignored. Examples:

```
-p
--subsp
```

`--org_excluded`

It specifies the taxonomic groups that are to be excluded from the analysis. Names must be provided according to the GenBank taxonomy and separated by spaces. If the name of a group consists of two or more words, these have to be joined by a plus '+' sign. This argument can be used with or instead of `--org_excluded_file`. Examples:

```
--org_excluded Daucinae
--org_excluded Daucinae Laserpitium+siler
```

`--org_excluded_file`

A path to the file containing a list of taxonomic groups that are to be excluded from the analysis (one per line). Names must be provided according to the GenBank taxonomy. This argument can be used with or instead of `--org_excluded`. Example:

```
--org_excluded_file excl.txt
```

Note: Excluding taxa from the analysis can be very useful if one needs to specify a paraphyletic taxon or does not want to waste computational resources for model organisms having hundreds of thousands sequences deposited in GenBank, e.g. corn or human.

`-d`, `--string_distance`

This parameter describes the number of allowed mismatches between two strings to consider them identical. This option has a strong effect on the algorithm joining ITS sequences based on descriptive features of GenBank records: specimen voucher, authors and publication. The parameter is by default set to 2 what compromises proper typo recognition and relatively low false positive joins rate. Examples:

```
-d 3
--string_distance 3
```

`--api_key`

A personal API key that identifies NCBI account. The information is passed to Biopython Entrez and providing a valid API key can speed up the analysis from three to ten requests per second. Example:

```
--api_key APIKEY
```

`--debug`

This argument prevents working folder named 'wf' from being deleted after the analysis has been finished. The folder contains among others FASTA files used for blasting and containing all sequences MatPhylobi processed searching for representative ones.

```
--debug
```

Details of algorithms and solutions

MatPhylobi's workflow consists of the following steps: 1) for each taxon and molecular marker, homological sequences to the reference accession are found using BLAST, 2) optionally, corresponding ITS1 and ITS2 sequences are joined, 3) if for a particular taxon and a given molecular marker more than one sequence was found, a representative one is chosen and 4) the representative sequences and reports are saved in output files.

IMPORTANT! Results returned by the program for large queries are not likely to be replicated due to ties breaking randomly in a situation when two or more sequences have equal scores and each can be considered as the best representative for a particular pair of taxon and molecular marker.

Downloading the data and BLAST queries

At the beginning of the analysis MatPhylobi takes the advantage of:

- Biopython's Entrez module for communication with NCBI Entrez tools and downloading the GenBank records
- Blastn tool form BLAST+ package for finding nucleotide sequences exhibiting local similarities to the query sequence.

However, depending on the localization of BLAST databases (localhost or remote), the MatPhylobi's pipeline is somewhat different.

IMPORTANT! Results of standalone and remote blastn can slightly differ due to differences in a way of limiting taxonomic scope of the search. In rare cases remote BLAST can return a taxon that should not be included in the analysis, e.g. is classified in a different genus than specified.

Processing of the BLAST results

BLAST output files contain identifiers of subject sequences, their total percentage query coverage and the strand in regard to the query sequence, taking into account assumption that the query sequence represents the coding (plus) strand. At this step, the sequences below user-provided query coverage threshold are excluded as well as sequences of hybrids, clones or those not identified to species level at all. Then, each accession from the list is annotated using Entrez eFetch query with additional information:

- taxon name according to GenBank Taxonomy,
- taxonomic identifier from GenBank Taxonomy and specimen voucher (if available),
- nucleotide sequence,
- length of the sequence neglecting unknown bases ('N's).

At this step, taxa names and taxonomic identifiers are compared to the list of synonyms provided by the user and changed accordingly. Sequences that are reported as representing non-coding (minus) strand are complementary reversed.

Recognizing and joining ITS1 and ITS2 sequences

Turning ITS joiner on by `-i` or `--its` option causes retrieval of additional information during eFetch query: 1) segment – ITS1 or ITS2 and 2) authors of the last reference appearing in the GenBank record. If the record does not contain information about segment, a special algorithm classifies the sequence as ITS1, ITS2 or 5.8S rDNA flanked by both ITS1 and ITS2 based on the pre-processed BLAST results containing information about the length of the subject sequence and its alignment to the query sequence. The whole process of joining ITS sequences consist of the following steps:

1. Every sequence is annotated with a segment (ITS1, ITS2 or both) based on explicit information from GenBank record or percentage query sequence coverage and alignment of the subject and query sequences.
2. ITS sequences which were recognized only as ITS1 or ITS2 are joined in pairs using the depth of the algorithm chosen by the user. If the option `-i` or `--its` was chosen ITS1 and ITS2

sequences will be joined only if they come from the same specimen voucher and published by the same authors. Adding one more ‘i’ (option `-ii` in effect) causes an attempt to join the remaining ‘free’ ITS1 and ITS2 using GenBank taxid and specimen voucher features. Using `-iii` option is a ‘brute force’ that joins the remaining sequences solely if they come from the same taxon.

3. If, according to the above rules, there is more than one possibility of joining sequences in pairs, sequences are first sorted by their accessions numbers and then joined. This reflects a common practice that sequences from different specimens are deposited in GenBank using consecutive numbers.
4. Misspellings in specimen voucher and publication GenBank record features are handled by ignoring the whitespaces and letter case when comparing strings. Additionally, Damerau-Levenshtein algorithm is used for calculation the distances (number of transformations) between two strings. By default two specimen voucher descriptions or references are considered identical if the calculated distance is less than three. However, the threshold can be set to a different value using `-d` argument. MatPhylobi utilizes an adapted to Python 3 implementation of Damerau-Levenshtein algorithm by Michael Homer (MIT, 2009), that can be found under this link: https://web.archive.org/web/20150909134357/http://mwh.geek.nz:80/2009/04/26/python-damerau-levenshtein-distance/

IMPORTANT! Two matching sequences are pasted using a repeat of 100 ‘N’, i.e. undefined nucleotides, as a separator. This makes identification of ITS1 and ITS2 boundaries much easier in the output FASTA file and does not affect negatively downstream application of multiple sequence alignment software.

Choosing the representative sequences

Problem of choosing a single representative sequence for particular taxon and molecular marker is solved by the following algorithm:

- If there is one sequence, it is chosen (trivial case).
- It there are two sequences, the longer one (neglecting the undetermined bases – N’s) is chosen.
- If there are three or more, blastn “all against all” is executed, annotating each sequence with the following parameters:
    - summed percentage identities to other sequences,
    - summed percentage query coverage of every other sequence.

  Next, the sequences are grouped in two (often overlapping) sets:
    - sequences having the sum of identities higher than 95% mean sum of identities within the group
    - sequences having the sum of query coverages higher than 95% mean sum of query coverages within the group.

If the intersection of the sets has one or more sequences, MatPhylobi chooses the sequence(s) with the highest sum of query coverage from them. If there is more than one such sequence, the algorithm chooses that having the highest sum of identities. If there is still more than one left – a random from the longest ones is chosen.

If the intersection is empty, MatPhylobi chooses sequences with the highest sum of identities from the second set, and then a random from the longest ones.

This algorithm ensures that the chosen sequence is “the most average”, i.e. the most similar to all remaining members of a group. More importantly, it throws out the outliers – presumably the sequences coming from misidentified specimens or having high rate of sequencing errors. However, it is worth notice that if only two sequences are taken into consideration, choosing the longest one is arbitrary and can lead to unexpected results if the longer sequence is contaminated.

Help and collaboration

In case of any questions, remarks or bugs please do not hesitate to contact us via matphylobi@gmail.com

In the case of reporting a bug, please provide us with an information on:

- versions of the operating system and the packages listed in dependencies
- config.ini file that MatPhylobi creates automatically for each run
- screenshot/text from the console containing the error message
- short description of the problem