

# TriLoNet User Manual

James Oldman

## 1 Introduction

**TriLoNet** is a Java software package presented in [1] created to construct rooted level-1 phylogenetic networks from aligned DNA sequence data. The software package consists of the main module:

- **TriLoNet** - constructs level-1 phylogenetic networks from a dense collection of trinets.

In addition, it also contains the following four complementary tools including:

- **SeqTrinet** - constructs a dense set of trinets from a multiple sequence alignment
- **TriExtract** - extracts the set of trinets from a level-1 phylogenetic network
- **PNDist** - calculates two distance measures for a pair of level-1 phylogenetic networks
- **PNGenerator** - randomly generates a set of  $n$  level-1 phylogenetic networks on  $l$  leaves

The algorithms and details for the last three tools can be found in [2].

## 2 Availability and Installation Requirements

**TriLoNet** 1.2 (and accompanying tools) have been implemented in Java and compiled using Java SE 8.0, please ensure Java Virtual Machine (JVM) version 8.0 or above has been installed [3]. Next, download the file **TriLoNet.zip** from <https://www.uea.ac.uk/computing/trilonet/> and extract the contents. The zip file contains this manual in PDF format, **TriLoNet.jar**, **TriExtract.jar**, **PNDist.jar** and **PNGenerator.jar**. These files are platform independent and can be run on Windows, Mac OSX and Linux. The instructions here assume a Mac OSX environment.

When running these tools on large datasets, it is recommended to allocate more RAM for the heap space, otherwise an **OutOfMemoryError** may occur. Heap space can be increased using the command line option **-Xmx**. For example, use the following command:

```
java -Xmx4096m -jar TriLoNet.jar
```

to allocate 4GB of heap space for use by **TriLoNet** (replace **TriLoNet** for **TriExtract**, **PNDist** or **PNGenerator** depending on the program).

## 3 Input Overview

**TriLoNet** primarily accepts three file formats as input; **.nex**, **.fsa** and **.tnets**. See [4] and [5] as well as some example files we have included for more information on the NEXUS and FASTA file formats. The reading in and outputting of eNewick strings [6] has been adapted from the **Network** class of **Lev1generator** [7], software developed for the generation of random level-1 phylogenetic networks and the FASTA file reading has been adapted from **FastaReader.java** [8]. If two or more sequences in the given input file are identical, **TriLoNet** will remove the duplicate sequences and relabel one of the sequences to represent the duplicate sequences. The taxa names are separated by two underscore characters (**\_**). For example, given either a NEXUS or FASTA file containing sequences {A, B, C, D, E, F, G} with taxa A, C and F being duplicates, after preprocessing, the set of sequences is {A\_C\_F, B, D, E, G}.

The TNETS format is simply a text file containing a dense collection of trinets, with exactly one trinet per line. Each line contains four pieces of information: the three leaf labels of the trinet and the trinet type. Please note that acceptable taxa labels include alphanumeric characters (A-Z, a-z, 0-9) and the underscore (\_) symbol. The hash symbol (#) is reserved for the labelling of reticulation vertices and the letter n followed by any number (e.g. n0 or n28) is reserved for the labelling of internal vertices. The label “root” is also reserved. Ensure that there are no spaces in the taxa labels and that all taxa labels are unique. See the included file `example.tnets` for an example of the TNETS format. The input file `example.tnets` contains the following 4 lines of text:

```
a c b S1
b a d N3
a c d T1
b c d N3
```

which correspond to the trinets displayed by the phylogenetic network presented in Figure 1.

## 4 Basic Usage

To get started, extract the contents of `TriLoNet.zip` and navigate to the location of the `TriLoNet.jar` file using the command prompt/terminal. To run `TriLoNet` from the command line and depending on the input (.NEXUS, .FSA or .TNETS), use

```
java -jar TriLoNet.jar example.nex
```

or

```
java -jar TriLoNet.jar example.fsa
```

or

```
java -jar TriLoNet.jar example.tnets
```

which will result in `myOutput.dot` (open in GraphViz) and `myOutput.txt` which contains the eNewick string of the constructed phylogenetic network as well as a summary of the construction process. The file `output.tree` will contain the eNewick string representation of the network constructed by `TriLoNet`. Additionally, the names of the output files can be specified when running `TriLoNet`, for example:

```
java -jar TriLoNet.jar example.nex anExample.dot exampleOutput.txt
```

## TriLoNet

`TriLoNet` can take in three types of input including aligned DNA sequence data (on the alphabet  $\{A, C, G, T\}$ ) in NEXUS or FASTA format, as well as a dense set of trinets in TNETS format. From this input, `TriLoNet` will construct and output a phylogenetic network in eNewick and DOT format. The command

```
java -jar TriLoNet.jar example.tnets myOutput.txt
```

will instruct `TriLoNet` to take as input the set of trinets contained in the file `example.tnets` and output the files `myOutput.txt` and `myOutput.dot`.

The text file `myOutput.txt` will contain a summary of the network construction process as well as the eNewick string of the network constructed by `TriLoNet`. If everything has installed correctly, the DOT file `myOutput.dot`

will contain a graphic representation of this phylogenetic network, which is shown in Figure 1. See the Viewing Output section for more information on the DOT format.

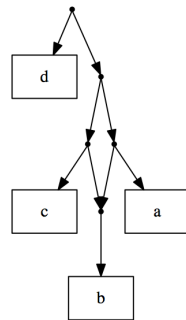


Figure 1: An example phylogenetic network.

## SeqTrinet

**SeqTrinet** will read in a multiple sequence alignment and output a dense collection of trinetets. To use the **SeqTrinet** module, use the argument `--st`. For example, the command

```
java -jar TriLoNet.jar Cooper.nex --st
```

will instruct **TriLoNet** to use **SeqTrinet** to output a dense collection of trinetets constructed from `Cooper.nex`.

## 5 Viewing Output

The Graph Visualisation program GraphViz [9] is required to view the `.dot` file representing the networks constructed by **TriLoNet**. This file will provide a visual representation of the network constructed by **TriLoNet**. A summary of each run will be outputted as a text file. This text file will contain the eNewick representation of the phylogenetic network constructed by **TriLoNet** as well as detailed information on the construction process. The last line in the output text file is an eNewick string that can be pasted and viewed in the software package Dendroscope [10].

## 6 Command line Options

The user can also optionally specify breakpoints when using a NEXUS or FASTA file as input with the argument `--b` followed by the desired sites separated by a comma. For example, the argument

```
--b10,15,18
```

would insert breakpoints at sites 10, 15 and 18 in a NEXUS or FASTA file.

The user can also modify the kappa threshold value used in the construction of a set of trinetets from a multiple sequence alignment in a NEXUS or FASTA file. The default value is 6.5. For example, to change this the user can specify the argument

```
--k4.0
```

to change the kappa threshold to 4.0.

## 7 Example

The website <http://phylonetworks.blogspot.co.uk/p/datasets.html> hosts several data sets containing recombinants. Here we have included an example on a giardia data set studied in [11]. In this data set the taxon 335 is a recombinant. The line

```
java -jar TriLoNet.jar Cooper.nex --b5979,7444
```

will result in the network shown in Figure 2.

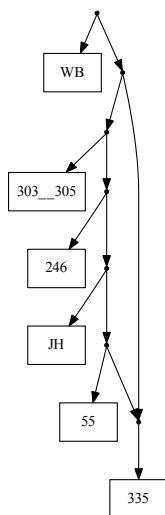


Figure 2: The phylogenetic network constructed by **TriLoNet** on a giardia data set studied in Cooper et al. (2007).

## TriExtract

The tool **TriExtract** will extract the trinets displayed by a level-1 phylogenetic network. To do this the user can specify an input .tree file containing an eNewick string and an output .tnets file to write to and record the set of trinets.

**TriExtract** accepts as input a single eNewick String from a .tree file. The reading in and outputting of eNewick strings [6] has been adapted from the Network class of Lev1generator [7], software developed for the generation of random level-1 phylogenetic networks. Please note that acceptable taxa labels include alphanumeric characters (A-Z, a-z, 0-9) and the underscore (\_) symbol. The hash symbol (#) is reserved for the labelling of reticulation vertices and the letter n followed by any number (e.g. n0 or n28) is reserved for the labelling of internal vertices. The label “root” is also reserved. Ensure that there are no spaces in the taxa labels and that all taxa labels are unique.

**TriExtract** outputs a tnets file. The tnets format is simply a text file containing a dense collection of trinets, with exactly one trinet per line. Each line contains four pieces of information: the three leaf labels of the trinet and the trinet type.

To get started, extract the contents of **TriExtract.zip** and navigate to the location of the **TriExtract.jar** file using the command prompt/terminal.

The eNewick string contained in **firstExample.tree** is:

$$(d, ((a, (b) \# H1), (c, \# H1)));$$

which corresponds to the phylogenetic network shown in Figure 1.

Entering

```
java -jar TriExtract.jar firstExample.tree output.dot output.tnets output.txt
```

into the terminal will instruct **TriExtract** to construct the network represented by the eNewick string contained in `firstExample.tree` and output the set of trinets displayed by this network to `output.tnets`. The file `output.dot` will contain a graphic representation of the network and `output.txt` will contain a summary of the extraction process. The only argument strictly necessary when running **TriExtract** is the `.tree` file, the arguments are used for specifying user selected names for the `.dot`, `.tnets` and `.txt` files created by **TriExtract**.

## PNDist

**PNDist** calculates two distance measures discussed in [2] between a pair of binary rooted level-1 phylogenetic networks. **PNDist** accepts as input a `.tree` file containing two eNewick strings on separate lines. See the **TriExtract** section for more information on eNewick strings and the `.tree` file format. **PNDist** calculates the  $Tn$  and  $Rf$  distance measure between the pair of networks given as input and prints the results to a `.txt` file.

As an example, entering

```
java -jar PNDist.jar example.tree
```

will produce a text file containing scores of  $D'tn = 0.6202$  and  $D'rf = 0.3448$ .

By default, **PNDist** will output both  $D'tn$  and  $D'rf$  to a `.txt` file. To calculate just one of the measures instead of both, the optional arguments `--tn` and `--rf` are used. For example, entering

```
java -jar PNDist.jar example.tree --tn
```

will instruct **PNDist** to calculate and output only  $D'tn = 0.6202$  to a `.txt` file.

## PNGenerator

**PNGenerator** randomly generates rooted binary level-1 phylogenetic networks and takes in two integers as input. One of the input arguments  $l$  specifies the number of leaves wanted in the generated network. The other input argument  $n$  specifies the number of random networks created by **PNGenerator**. To get started, extract the contents of **PNGenerator.zip** and navigate to the location of the **PNGenerator.jar** file using the command prompt/terminal.

As an example, entering

```
java -jar PNGenerator.jar --l50 --n100
```

into the terminal will instruct **PNGenerator** to generate 100 random networks each on 50 leaves. There is also the optional argument  $t$  that instructs **PNGenerator** to output trees instead of networks. For example, entering

```
java -jar PNGenerator.jar --l30 --n40 --t
```

into the terminal will instruct **PNGenerator** to generate 40 random trees each on 30 leaves.

**PNGenerator** will output the eNewick strings of the randomly generated networks (trees) to a `.txt` file.

## 8 Disclaimer

This software is supplied as-is, with no warranty of any kind expressed or implied. We have made every effort to avoid errors in design and execution of this software, but we will not be liable for its use or misuse. The user is solely responsible for the validity and consequences of any results generated. This program is distributed in the hope that it will be useful, but without any warranty; without even the implied warranty of merchantability or fitness for a particular purpose. See the GNU General Public License for more details.

## 9 Research Team

James Oldman, Prof. Vincent Moulton and Dr. Taoyang Wu

For technical questions, please e-mail [james.oldman@gmail.com](mailto:james.oldman@gmail.com)

TriLoNet is Copyright 2011-2015 James Oldman, Vincent Moulton and Taoyang Wu.

## References

- [1] James Oldman, Taoyang Wu, Leo van Iersel, and Vincent Moulton. Trilonet: Piecing together small networks to reconstruct reticulate evolutionary histories. *Molecular Biology and Evolution*, 2016.
- [2] Vincent Moulton, James Oldman, and Taoyang Wu. Comparing level-1 networks by counting trinets. *Preprint*, 2016.
- [3] Sun Microsystems. Java downloads for all operating systems. <http://java.com/en/download/manual.jsp>, 2015. [Online; accessed 5-Oct-2015].
- [4] Ecology and Evolutionary Biology Department of the University of Connecticut. Phylogenetics: Nexus format. [http://hydrodictyon.eeb.uconn.edu/eebedia/index.php/Phylogenetics:\\_NEXUS\\_Format](http://hydrodictyon.eeb.uconn.edu/eebedia/index.php/Phylogenetics:_NEXUS_Format), 2014. [Online; accessed 5-Oct-2015].
- [5] Zhang Lab University of Michigan. What is fasta format? <http://zhanglab.ccmb.med.umich.edu/FASTA/>, 2015. [Online; accessed 5-Oct-2015].
- [6] Monique M Morin and Bernard ME Moret. Netgen: generating phylogenetic networks with diploid hybrids. *Bioinformatics*, 22(15):1921–1923, 2006.
- [7] Katharina T Huber, Leo Van Iersel, Steven Kelk, and Radoslaw Sucheccki. A practical algorithm for reconstructing level-1 phylogenetic networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 8(3):635–649, 2011.
- [8] Joe Wandy. Parsing a fasta file in java. <https://raw.githubusercontent.com/joewandy/BioinfoApp/master/src/com/joewandy/bioinfoapp/model/core/io/FastaReader.java>, 2011. [Online; accessed 15-May-2014].
- [9] Emden Gansner, Eleftherios Koutsofios, and Stephen North. Drawing graphs with dot. 2006.
- [10] Daniel H Huson and Celine Scornavacca. Dendroscope 3: an interactive tool for rooted phylogenetic trees and networks. *Systematic biology*, page sys062, 2012.
- [11] Margarethe A Cooper, Rodney D Adam, Michael Worobey, and Charles R Sterling. Population genetics provides evidence for recombination in giardia. *Current Biology*, 17(22):1984–1988, 2007.