

Phyutility manual (v. 2.2)

Stephen A. Smith

November 20, 2007



Contents

1	Introduction	2
2	Installation	3
2.1	Download	3
2.2	Mac	3
2.3	Linux	4
2.4	Windows	4
3	Tree Functions	5
3.1	Rerooting	5
3.1.1	Examples	5
3.2	Pruning	5
3.2.1	Examples	5
3.3	Type conversion	6
3.3.1	Examples	6
3.4	Consensus	6
3.4.1	Examples	6
3.5	Leaf stability	7
3.5.1	Examples	7
3.6	Branch Attachment Frequency	7
3.6.1	Examples	7
3.7	Tree support	8
3.7.1	Examples	8
3.8	Thinning trees	8
3.8.1	Examples	8
4	Data matrix functions	9
4.1	Concatenate	9
4.1.1	Examples	9
4.2	GenBank Parsing	9

4.2.1	Examples	9
4.3	Trimming sites	10
4.3.1	Examples	10
4.4	Searching NCBI	10
4.4.1	Examples	10
4.5	Fetching Sequences from NCBI	10
4.5.1	Examples	11
5	References	11

1 Introduction

Phyutility (fyoo-til-i-te) is a program that collects many of the analyses and functions that phylogeneticists perform on trees and molecular data. Some of these were developed and are first introduced in this program, others may be found in other programs but may not be flexible for pipelining or different input and output formats. Currently phyutility performs

- Trees
 - rerooting
 - pruning
 - type conversion
 - consensus
 - leaf stability
 - lineage movement
 - tree support
 - thinning trees
- Data matrices
 - concatenate alignments
 - genbank parsing
 - trimming alignments
 - search NCBI
 - fetch NCBI

In order to get help from the command line just enter phyutility with no arguments (or `java -jar phyutility.jar`) and the list of commands will show. You may then enter `phyutility -h <nameofcommand>` (or `java -jar phyutility.jar -h <nameofcommand>`) for help on that command. Phyutility is an open source project that makes use of not only custom code, but also JADE (from the PEBLS library <http://code.google.com/p/pebls>) and JEBL (<http://sourceforge.net/projects/jeb1>). It additionally employs Derby (<http://db.apache.org/derby/>) for large tree files. When Derby is employed a folder named `readtrees` will be created and deleted. Make sure to not have a folder named `readtrees` beforehand or it will be deleted.

This manual explains installation, how each function works, the options for each, and a simple example. Report bugs here <http://code.google.com/p/phyutility/issues/list>

2 Installation

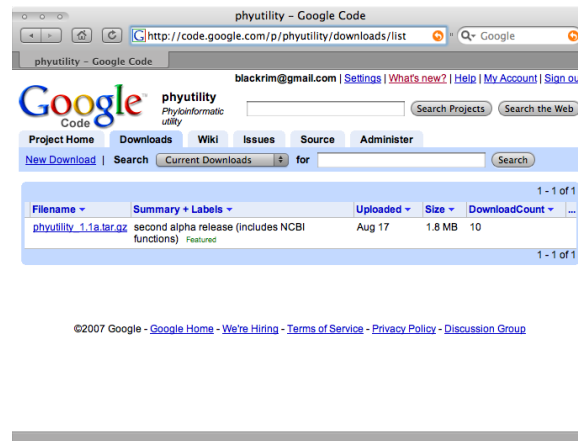
Phyutility is available for all operating systems that run Java. If your computer does not have Java installed, you need to install it. Mac users with versions ≥ 10.4 are fine (not sure about 10.3, but could be fine). Linux users should verify that they are using Sun Java and not the Gnu Java. Windows users should open the command prompt and verify that something happens when you type "java <enter>", if nothing happens, go here <http://www.java.com/en/download/index.jsp> and follow the instructions.

2.1 Download

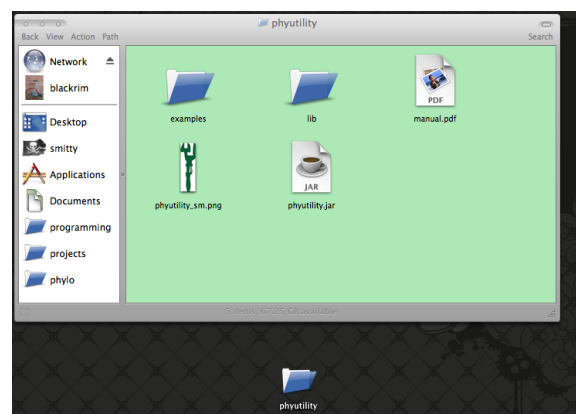
You can download the most recent package from <http://code.google.com/p/phyutility/downloads/list>. The package will include this documentation, a lib folder with the Derby package, example files and phyutility.jar. phyutility.jar is the actual program. you may run any of the procedures using phyutility.jar. These will be run from the terminal. Even though phyutility.jar can be run from any operating system that can run Java, below I have custom instructions for major operating systems to make life easier.

2.2 Mac

1. download package



2. unpack package (just double click it, it will make a folder called phyutility)



3. move package somewhere for permanance (here I have placed the folder into a folder, phylo, in my Applications folder)

Once you download the phyutility package and unpack, you may place the folder anywhere. It is easiest in Windows to simply work out of that folder and run examples as they are shown in this documentation. By work out of that folder, I mean, copy or drag your input files into the phyutility folder, then navigate to that folder in the Command Prompt and run your commands.

3 Tree Functions

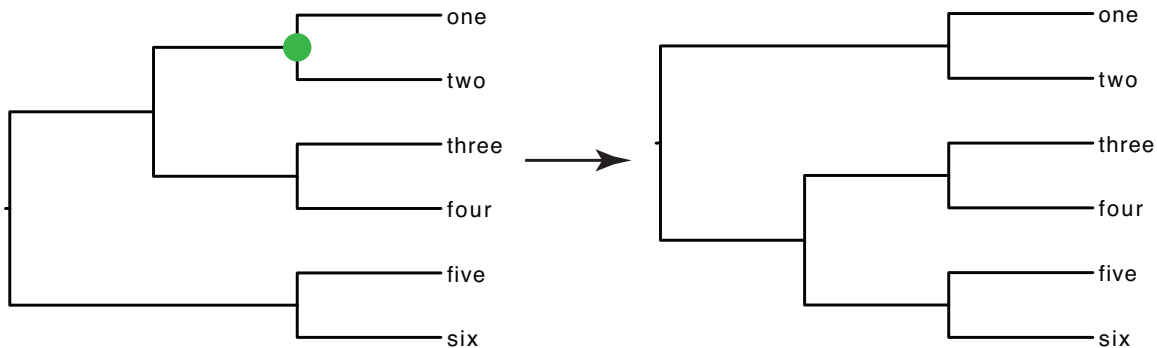
3.1 Rerooting

The need to reroot trees or a tree is common. Phyutility will reroot nexus or newick files and will output nexus or newick files. Rerooting can be done with unrooted or rooted trees and when multiple trees are in the file or multiple files are given, all the trees will be rerooted and placed in one file. **If no names are given to reroot, phyutility will unroot trees.**

3.1.1 Examples

Command-line options

- `-rr` | designates that you want to reroot
- `-names <tip name> ...` | tip names forming the mrca for the reroot
- `-in <file name> ...` | input tree files
- `-out <file name>` | output tree file name (optional)
- `-log <file name>` | log file name (optional)



```
java -jar phyutility.jar -rr -in test.tre -out testrr.tre -names one two
```

3.2 Pruning

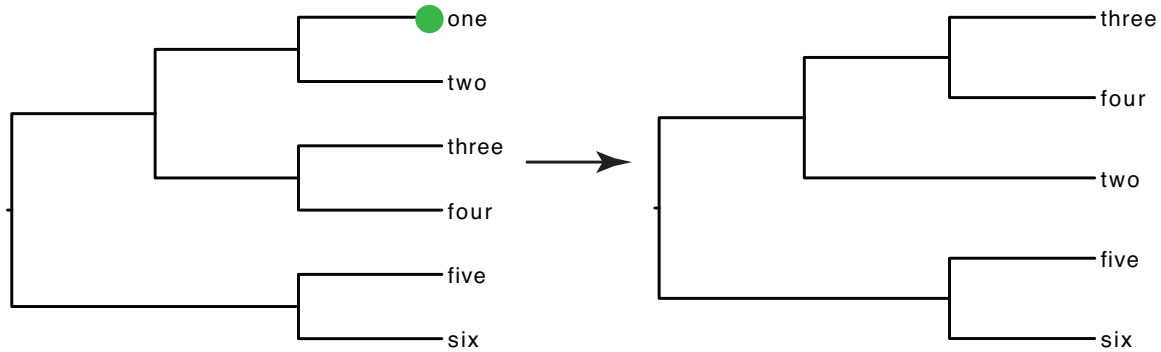
It is not uncommon to want or need to prune taxa from a tree. Often it is best to rerun the analyses with the correct taxa, but it is also helpful to be able to prune taxa without rerunning analyses. Phyutility will prune nexus or newick files and will output nexus or newick files. When multiple trees are in the file or multiple files are given, all the trees will be pruned and placed in one file.

3.2.1 Examples

Command-line options

- `-pr` | designates that you want to prune
- `-names <tip name> ...` | tip names to prune

- -in <file name> ... | input tree files
- -out <file name> | output tree file name (optional)
- -log <file name> | log file name (optional)



```
java -jar phyutility.jar -pr -in test.tre -out testpr.tre -names one
```

3.3 Type conversion

Often it is necessary to use tree files that have or do not have translation tables, or to use or not use newick or nexus tree files. Phyutility can convert between these types. Type conversion only occurs one file at a time so if multiple files are given, only the first will be used.

3.3.1 Examples

Command-line options

- -vert | designates that you want to convert
- -in <file name> | input tree file
- -out <file name> | output tree file name
- -log <file name> | log file name (optional)

```
java -jar phyutility.jar -vert -in test.tre -out testvert.nex
```

3.4 Consensus

Many programs make consensus trees, and phyutility is meant to only provide another convenient way to produce consensus trees from multiple (or single) file sources. Phyutility will make a consensus with nexus and/or newick files and will output nexus or newick files. When multiple trees are in a file or multiple files are given, all the trees will be used and a consensus will be placed into one out file.

3.4.1 Examples

Command-line options

- -con | designates that you want to make a consensus
- -t <number> | the threshold for consensus (1.0 = strict, 0.5 = majrule, 0 = allcompat)
- -in <file name> ... | input tree files

- -out <file name> | output tree file name
- -log <file name> | log file name (optional)

```
java -jar phyutility.jar -con -t 0.5 -in testall.tre -out test.con
```

3.5 Leaf stability

The leaf stability index was described and implemented in a Mac OS 9 program by [Thorley and Page \(2000\)](#). Phyutility implements this procedure and reports the leaf stability indices for all taxa in a tree or set of trees. Phyutility will calculate these with nexus and/or newick files. When multiple trees are in the file or multiple files are given, all the trees will be used.

3.5.1 Examples

Command-line options

- -ls | designates to perform the leaf stability procedure
- -in <file name> ... | input tree files
- -log <file name> | log file name (optional)

```
java -jar phyutility.jar -ls -in testall.tre
```

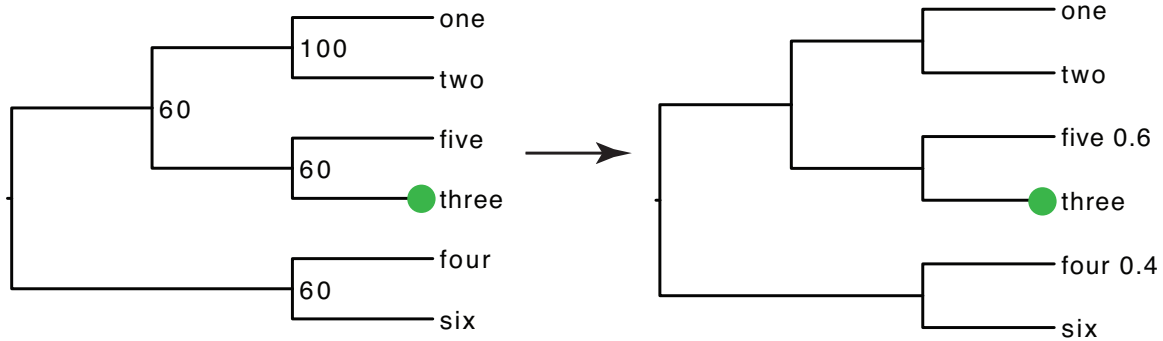
3.6 Branch Attachment Frequency

The lineage movement procedure is aimed at identifying where a lineage (tip or clade) moves. When there is a source set of trees (i.e. posterior distribution of trees from mrbayes) and a consensus tree with a target clade with lower support, the lineage movement procedure will plot where else the lineage is falling (as seen on the consensus tree). With a consensus tree with otherwise good support this can be very illuminating as to why and where there is low support, for consensus trees with otherwise low support, this will probably not be helpful. Phyutility will use nexus or newick files and will output nexus or newick files. When multiple trees are in the file or multiple files are given, all the trees will be used.

3.6.1 Examples

Command-line options

- -lm | designates that you want to run a branch attachment (lineage movement) analysis
- -names <tip name> ... | tip names to check
- -tree <file name> | consensus file to map movement
- -in <file name> ... | input tree files
- -out <file name> | output tree file name
- -log <file name> | log file name (optional)



```
java -jar phyutility.jar -lm -in testall.tre -tree test.con -out testlm.tre -names three
```

3.7 Tree support

A consensus tree is helpful in that it gives us an idea of what the best supported tree is in a set of trees. It can also be helpful to see what support there is for clades in a particular tree that may not be a consensus tree. For this, one can use the tree support function in Phyutility. Phyutility will use nexus or newick files and will output nexus or newick files. When multiple trees are in the file or multiple files are given, all the trees will be used and the support tree will be placed in one file. (In the case where the consensus tree is used for the tree for support, the same results should occur if you made a consensus tree).

3.7.1 Examples

Command-line options

- `-ts` | designates that you want to calculate tree support
- `-tree <file name>` | tree to get support for
- `-in <file name> ...` | input tree files
- `-out <file name>` | output tree file name
- `-log <file name>` | log file name (optional)

```
java -jar phyutility.jar -ts -in testall.tre -tree test.con -out testts.tre
```

3.8 Thinning trees

Trimming (or thinning) trees can be essential if other programs require less trees than are present in your files. Phyutility will thin these files to make them more manageable.

3.8.1 Examples

Command-line options

- `-tt #` | designates that you want to thin and followed by the number of trimming (sample every #)
- `-in <file name> ...` | input tree files
- `-out <file name>` | output tree file name
- `-log <file name>` | log file name (optional)

```
java -jar phyutility.jar -tt 100 -in testall.tre -out testts.tre
```

4 Data matrix functions

4.1 Concatenate

Concatenating alignments together is a necessary task but can often be a painful assignment, especially when the same taxa are not found in all alignments. Phyutility will concatenate fasta or nexus files. Sequences will be concatenated as long as they have the same name between files. Each input file is considered a gene. Missing taxa in each gene will be given gaps for each gene in which the taxon is missing. The output is nexus or fasta and each needs to have been aligned separately before running.

4.1.1 Examples

Command-line options

- `-concat` | designates that you want to concatenate
- `-in <file name> ...` | input fasta or nexus files, each one considered a gene
- `-out <file name>` | output file name
- `-log <file name>` | log file name (optional)

```
java -jar phyutility.jar -concat -in test.aln test2.aln -out testall.aln
```

4.2 GenBank Parsing

When downloading many sequences from GenBank in a fasta file, it can be helpful to parse the id line with a more helpful identifier. Phyutility will do this with many options:

1. gi number
2. gb number
3. taxon name
4. taxon_name
5. T_name
6. taxon_name_ginumber
7. taxon_name_gbnumber

The input must be fasta (aligned or unaligned) and the output should always be double-checked as some GenBank entries have non-standard id lines.

4.2.1 Examples

Command-line options

- `-parse #` | designates that you want to parse and the option number must follow
- `-in <file name> ...` | input fasta files
- `-out <file name>` | output file name
- `-log <file name>` | log file name (optional)

```
java -jar phyutility.jar -parse 1 -in test.gb -out testgb1.fasta
```

4.3 Trimming sites

Many people (most people) edit their alignments by hand, but would often like to have some standard way to delete sites with gaps. Many alignment programs allow this but they often don't allow you to choose what threshold to delete sites. In other words, you may want to delete sites that are missing 50% and 75% data. Phyutility will do this. The input is fasta or nexus. Trimming is done one file at a time, so if multiple files are given, only the first will be read.

4.3.1 Examples

Command-line options

- `-clean #` | designates that you want to trim and the threshold must follow
- `-in <file name>` | input file
- `-out <file name>` | output file name
- `-log <file name>` | log file name (optional)

```
java -jar phyutility.jar -clean 0.5 -in test.nex -out test50.nex
```

4.4 Searching NCBI

NCBI has a very convenient web interface for searching for particular things such as nucleotide sequences in GenBank. The NCBI searching function in phyutility is meant not to replace this functionality but instead to offer a way to double check a fetch request (see below). Basically to verify that the fetch you are about to do returns the expected ids and count number.

4.4.1 Examples

Command-line options

- `-es` | designates that you want to perform a search
- `-term <term>` | the search term(s)
- `-db #` | corresponds to the database you want to search (right now nucleotide (1), protein (2), genome (3), taxonomy (4)) (optional, default = nucleotide)
- `-log <file name>` | log file name (optional)

```
java -jar phyutility.jar -es -term lonicera+OR+viburnum+AND+rbcl -log log.txt
```

4.5 Fetching Sequences from NCBI

NCBI has a very convenient web interface for fetching sequences as well. However, it is occasionally beneficial, instead of using something like the parse function (above) to just parse the results at the outset. The NCBI fetch function in phyutility is meant not to replace the web interface, just to offer another route to retrieving fasta sequences from GenBank. This is also a first step to add even more post-processing of genbank retrievals. The output is a fasta file formatted as requested.

4.5.1 Examples

Command-line options

- `-ef` | designates that you want to perform a fetch
- `-term <term>` | the search term(s)
- `-out <file name>` | output file name
- `-ll <length>` | the length limit to return, meant to eliminate genomic sequences or sequences that are too long in the retrieval (should almost always use, especially use if you get a memory error) (optional, default = 10000)
- `-outfor <format>` | a way to customize the fasta line (1 = ginumber, 2 = taxid, 3 = orgname (with spaces replaced with whatever is sep), 4 = define (with spaces replaced with sep), 5 = seqlength) (optional, default = 31)
- `-sep <separator>` | separator for between outfor and between orgname and define (optional, default = `_`)
- `-db #` | corresponds to the database you want to search (nucleotide (1), protein (2)) (optional, nucleotide is default)
- `-log <file name>` | log file name (optional)

```
java -jar phyutility.jar -ef -term lonicera+OR+viburnum+AND+rbcl -ll 3000  
-log log.txt -out out.fasta -sep -outfor 13
```

Acknowledgments

Thanks to all the testers and to Brian Moore for the name.

5 References

Thorley, J. L. and R. D. Page. 2000. Radcon: phylogenetic tree comparison and consensus. *Bioinformatics* 16:486–487.