TreeCollapseCL 3.2

Emma Hodcroft
Andrew Leigh Brown Group
Institute of Evolutionary Biology
University of Edinburgh

This command-line Java program takes in Nexus/Newick-style phylogenetic tree code that includes branch lengths and may include bootstrap values, and analyses them using any combination of three main functions:

- Rooting the tree
- Collapsing the tree
- Measuring the length and other attributes of the tree

More details of these three functions are provided below. The program output varies by the functions selected, but trees are output in Nexus/Newick-style code, and length attributes are output in a CSV (comma delimited) file.

This program accepts two types of file - to see details on these types, please see '`-t`' below.

PLEASE NOTE: I am not responsible for any incorrect behaviour of this program. I do not guarantee that it will behave correctly or as you predict because I cannot test it in every conceivable situation.

More details on the main three functions of the program:

- Rooting the tree
  An outgroup is specified by the user, and the tree is rooted or re-rooted to that outgroup. The rooting function behaves in the same way as the FigTree (http://tree.bio.ed.ac.uk/software/figtree/) program, so that after rooting, the branch length of the branch parental to the outgroup is split evenly between the outgroup and the rest of the tree, and the new root node has no bootstrap value.
- Collapsing the tree
  A threshold is provided by the user, and all nodes with bootstrap values below this threshold are collapsed to polytomies. Length of the tree is preserved.
- Measuring length and other attributes
  The length from each leaf to the node above the root node (if an outgroup is present, the length from root to first node can be dictated more by the outgroup than anything else) is calculated, and the average bootstrap value (average of each bootstrap value between the leaf node and the root) is calculated for each leaf.

Updates since version 3.1.2: (Previous updates on the last page)

- Now compatible with Windows Shell auto-glob when passing * to specify file endings
- Removed outputting the 'level' of a node when specifying the '`-l`' parameter
- '`-t`' is now optional, if not included the value defaults to '`0`'
- '`-rax`' is now depreciated (but can be included without affecting the run). The program now automatically detects and handles a larger number of minor variations in format that can occur at the end of Newick files
- Corrected error that assumed all files had a path before the file name (caused a StringIndexOutofBoundsException)
- Corrected a minor collapsing problem that caused slight under-collapsing of some nodes
- Corrected rooting to be more robust with particular polytomies

The parameters '-d' *or* '-f' MUST be supplied. The parameters '-b', '-l', '-t', '-p', '-v', '-nbs', and '-r' *or* '-rp' are optional. The parameter '-rax' is now not needed and is depreciated.

```
java -jar TreeCollapseCL3.jar -t
                                -d  dir or -f file
                               [-b] bootstrapThreshold
                               [-r] outgroup
                               [-rp] outgroup
                               [-v]
                               [-l]
                               [-p]
                               [-nbs]
                               [-rax](depreciated)
```

If the program is run without any parameters, a list of parameters with descriptions is displayed.

A list of example runs using various combinations of the three main functions can be found at the end of this document, as well as some tips on running multiple files, and citation information.

Parameters:

`-b bootstrapThreshold` (optional)

Use '-b' to indicate that you would like to collapse the tree by specifying the bootstrap threshold to be used to collapse the nodes. Nodes at or below this threshold will be collapsed. This will be done after rooting the tree, if rooting has also been specified. If calculating the length and other attributes has been specified, specifying '-b' will mean that 'level' is included in the length file (see '-l' for more details).
'-b' can be specified as a decimal or an integer, depending on the type of bootstraps your file contains.
Example: `java -jar TreeCollapseCL.jar -b 50`
Sets threshold at 50, perhaps appropriate for bootstraps ranging from 0 to 100
Example: `java -jar TreeCollapseCL.jar -b 0.5`
Sets threshold at 0.5, perhaps appropriate for bootstraps ranging from 0 to 1.0
You can also specify a range of bootstrap thresholds to be used by putting them in parenthesis separated by commas (no spaces!). Each tree will be run with all thresholds, and output in a file that reflects the threshold used.
Example: `java -jar TreeCollapseCL.jar -b (0.5,0.7) -t O -f Sequences.newick`
Runs files at thresholds 0.5 and 0.7, will output files called '`Sequences_0.5coll.newick`' and '`Sequences_0.7coll.newick`'
To force the filename to exclude the decimal point ('.') in decimal thresholds, see '-p' parameter

`-t` (optional)

Use '-t' to specify the file type that will be read in. If this parameter is not given, it defaults to 'O'.
-t O (Default) Use 'O' (capital o) to specify that the file is the usual Newick/Nexus-type file, with bootstrap values preceding colons. Unless the file was exported as Nexus with annotation in FigTree, use this option (or don't include the parameter at all)!
Example: "`((B:0.04,C:0.03)0.83:0.01);`" Where 0.83 is the bootstrap value.
-t F Use 'F' to specify that the file is a Nexus-type file that's been exported from FigTree with annotations. These have bootstrap values within square brackets ('[]').
Example: "`((B:0.04,C:0.03)[&bs=0.83]:0.01);`" Where 0.83 is the bootstrap value.

`-d dir` *or* `-f file`

These specify the file (use '`-f`') or directory (use '`-d`') of files to be read in.
Follow '`-f`' with the file name.
Example: `java -jar TreeCollapseCL.jar -b 0.5 -t O -f Sequences.newick`
Follow '`-d`' with the directory containing the files to be read in. `%CD%` can be used as well. If there may be spaces in folders or filenames in the path, use double quotes ("") to enclose the path. It's a good idea to use these unless you're certain there are no spaces.
Example: `java -jar TreeCollapseCL.jar -b 0.5 -t O -d C:\Users\Bob\Sequences`
Example: `java -jar TreeCollapseCL.jar -b 0.5 -t O -d "%CD%"`
You can also specify the ending of the files to be read by using '*.' followed by the ending. Be aware that this will only work on endings - putting something before the * will not work.
Example: `java -jar TreeCollapseCL.jar -b 0.5 -t O -d C:\Users\Sequences\*.newick`
Example: `java -jar TreeCollapseCL.jar -b 0.5 -t O -d "%CD%\*.nexus"`

`-p` (optional)

Use this to specify that output file names should not include the decimal point ('.') in the decimal thresholds when multiple bootstrap thresholds are provided.
Example: `java -jar TreeCollapseCL.jar -b (0.5,0.7) -t O -f Sequences.newick`
Will output files called '`Sequences_0.5coll.newick`' and '`Sequences_0.7coll.newick`'
Example: `java -jar TreeCollapseCL.jar -b (0.5,0.7) -p -t O -f Sequences.newick`
Will output files called '`Sequences_05coll.newick`' and '`Sequences_07coll.newick`'
(Using this parameter on integer thresholds will not change anything.)

`-l` (optional)

Use this to specify that length and other attributes of the tree should be calculated. A different length file will be output for each input tree and each collapsing threshold, if appropriate. Output is in a CSV (comma delimited) file with three columns corresponding to leaf name, length from that leaf to the node previous to the root, and average bootstrap value from that leaf to the root. If the user has specified the rooting option ('`-r`' or '`-rp`'), the length and average bootstrap will be calculated after rooting.
Example: `java -jar TreeCollapseCL.jar -b 0.5 -p -l -t O -f Sequences.newick`
Will output files called '`Sequences_05coll.newick`' and '`Sequences_05coll.csv`'

`-r outgroup` *or* `-rp outgroup` (optional)

Use this to specify that the tree should be rooted by the specified outgroup. `Outgroup` can be specified by including the name of a single node or the name of a file that contains a list of the outgroup nodes, one name per line, no punctuation. Specifying '`-rp`' will root the tree and also output a Nexus/Newick-style copy of the rooted tree (recommended), whereas '`-r`' will root the tree without any additional output. Rooting is carried out before calculating length and other attributes and before collapsing (if either of these is specified). **All input files in a run will be rooted by the same outgroup!**
Example: `java -jar TreeCollapseCL.jar -b 0.5 -t O -p -rp G101360 -f Sequences.newick`
Will root by the node 'G1010360' and output files called '`Sequences_05coll.newick`' and '`Sequences_root.newick`'
(Continued…)

Example: `java -jar TreeCollapseCL.jar -b (0.5,0.7) -t O -p -l -r roots.txt -f Sequences.newick`

Will root by the node nodes listed in the file 'roots.txt' and output files called '`Sequences_05coll.newick`', '`Sequences_07coll.newick`', '`Sequences_05coll.csv`', and '`Sequences_07coll.csv`' but no rooted tree file.

`-nbs` (optional)

Use this to specify that any output tree files should NOT contain bootstrap values. The original tree will also be returned without bootstraps. If collapsed or rooted trees are also being returned, they will lack bootstrap values. This can be useful if the user plans to use some functions in the '`R`' packages '`ape`' or '`MCMCglmm`,' as bootstrap information can cause some functions to work incorrectly.

Example: `java -jar TreeCollapseCL.jar -b 0.5 -p -nbs -t O -f Sequences.newick`

Will output files called '`Sequences_05coll.newick`' and '`Sequences_nbs.newick`' – neither will contain bootstrap values

`-rax` (depreciated)

This used to be required to specify the specific format of a Newick file. The program now detects this automatically so this parameter is not needed anymore. For backwards compatibility, you can include it as a parameter, but it will not affect the run.

`-v` (optional)

Use this to turn on very crude 'debug' which will basically output intermediate flags and steps to the console. It's probably not very useful, and will significantly slow down runtime and possibly even crash the run if turned on for very large files. It is recommended that the user create a smaller 'test' file if they wish to turn on the debug option. Again, it's probably not very useful.

(Examples of program runs on the next page…)

Examples of program runs:

These examples assume the user always wants to exclude decimal points ('-p') from output files and is always using the 'other' file type ('-t O'). Change these settings as needed.

Remember that for input, a directory ('-d') can be specified instead of a file ('-f'), multiple bootstraps can be used ('-b (0.5,0.7)') instead of just one ('-b 0.5'), output trees and a copy of the original tree can be returned without bootstrap values ('-nbs'), and for an outgroup either a node name ('-r G101360') or a file containing multiple node names ('-r roots.txt') can be used.

To find length of a tree:
```
java -jar TreeCollapseCL.jar -t O -p -l -f Sequences.newick
```

To root the tree and print the rooted tree:
```
java -jar TreeCollapseCL.jar -t O -p -rp roots.txt -f Sequences.newick
```

To root and find length:
```
java -jar TreeCollapseCL.jar -t O -p -l -r roots.txt -f Sequences.newick
```

To root, find length, and collapse:
```
java -jar TreeCollapseCL.jar -b 0.5 -t O -p -l
                                    -r roots.txt -f Sequences.newick
```

To find length and collapse:
```
java -jar TreeCollapseCL.jar -b 0.5 -t O -p -l -f Sequences.newick
```

To collapse only:
```
java -jar TreeCollapseCL.jar -b 0.5 -t O -p -f Sequences.newick
```

To root and collapse:
```
java -jar TreeCollapseCL.jar -b 0.5 -t O -p -r roots.txt -f Sequences.newick
```

Tips on running multiple files:

If the user needs to run multiple files as input (by specifying a directory ('-d')), ensure they all have different names. If they have been generated by a batch run or are otherwise likely to have similar names, this program is designed to handle files that are numbered with the number in between decimal points preceding the file ending, as shown:

```
SeqSet_run.1.newick
SeqSet_run.2.newick
```

In order for this to work correctly when specifying that length be calculated ('-l'), your input files should all have a file ending that begins with '.n'. For this purpose, '.newick', '.nexus', '.nex', or even just '.n' files will work.

Citation:

If you publish or present work that has been processed using this program, please cite Emma Hodcroft and the website where this program can be downloaded (http://emmahodcroft.com/TreeCollapseCL3.html).

Updates from Previous Versions:

Updates since version 3.1.1:

- Corrected file/directory reading for Unix/Linux/Mac users

Updates since version 3.1:

- Please read updated information on when and how to use '`-rax`' and '`-t`'!
- Now takes trees with FigTree annotation or in Newick format without bootstrap values, or with some nodes missing bootstrap values (collapsing cannot be done if bootstrap values are not present)
- Corrected errors in reading Nexus files
- Corrected errors in handling file names that included decimal points

Updates since version 3.0:

- Now takes trees without bootstrap values, or with some nodes missing bootstrap values (collapsing cannot be done if bootstrap values are not present)
- Can now correctly handle trees output from RAxML (or other programs) where an extra root branch length of '0.0' has been attached to the end of the code (see parameter '`-rax`')
- Stops the run and provides an error message with potential solutions if too much of the tree has been collapsed, causing a recursive stack overflow