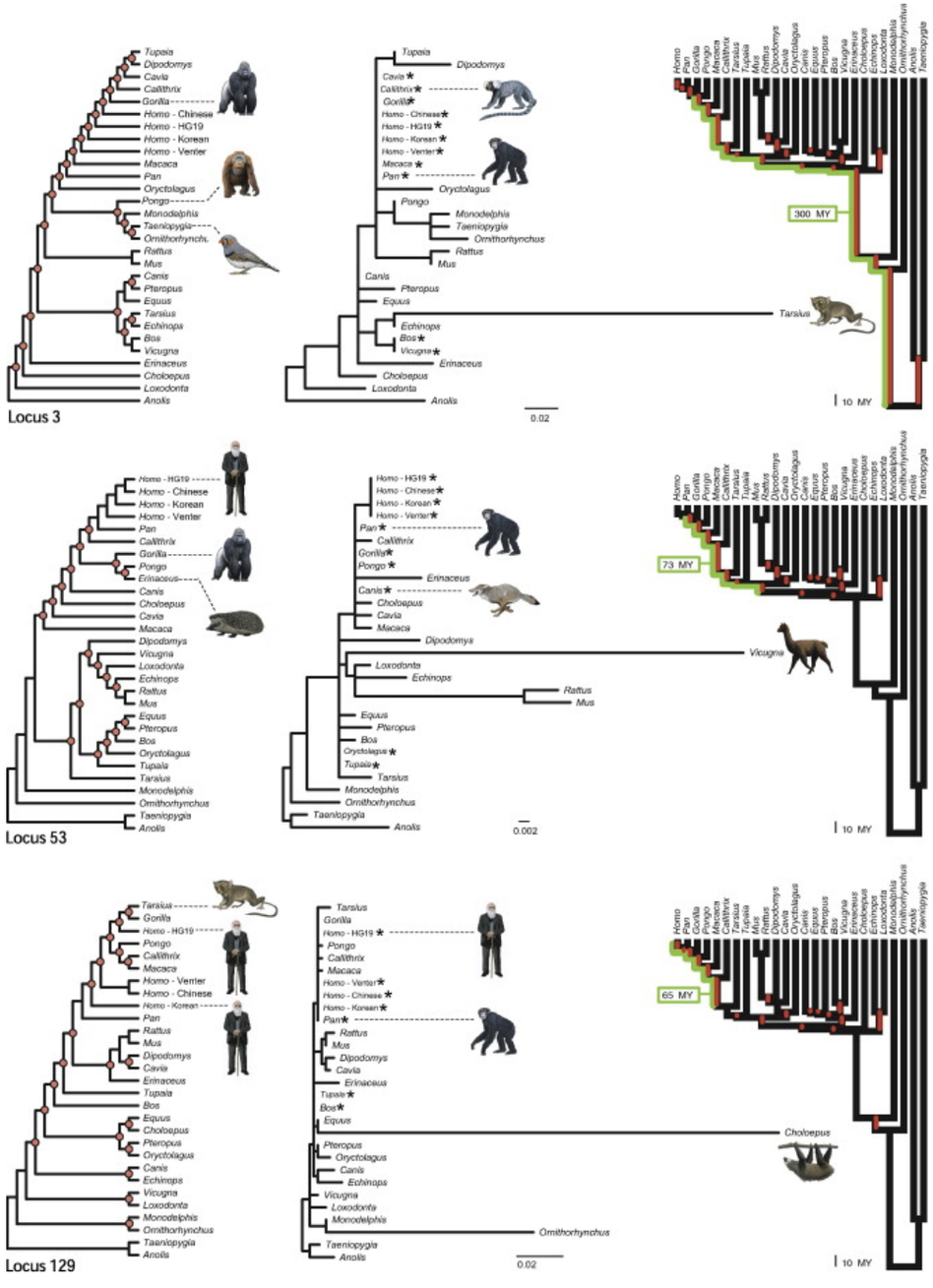
Finding Errors in Phylogenomic Data Using TreeShrink

Uyen Mai
University of California San Diego
umai@eng.ucsd.edu

TreeShrink Software
https://github.com/uym2/treeshrink

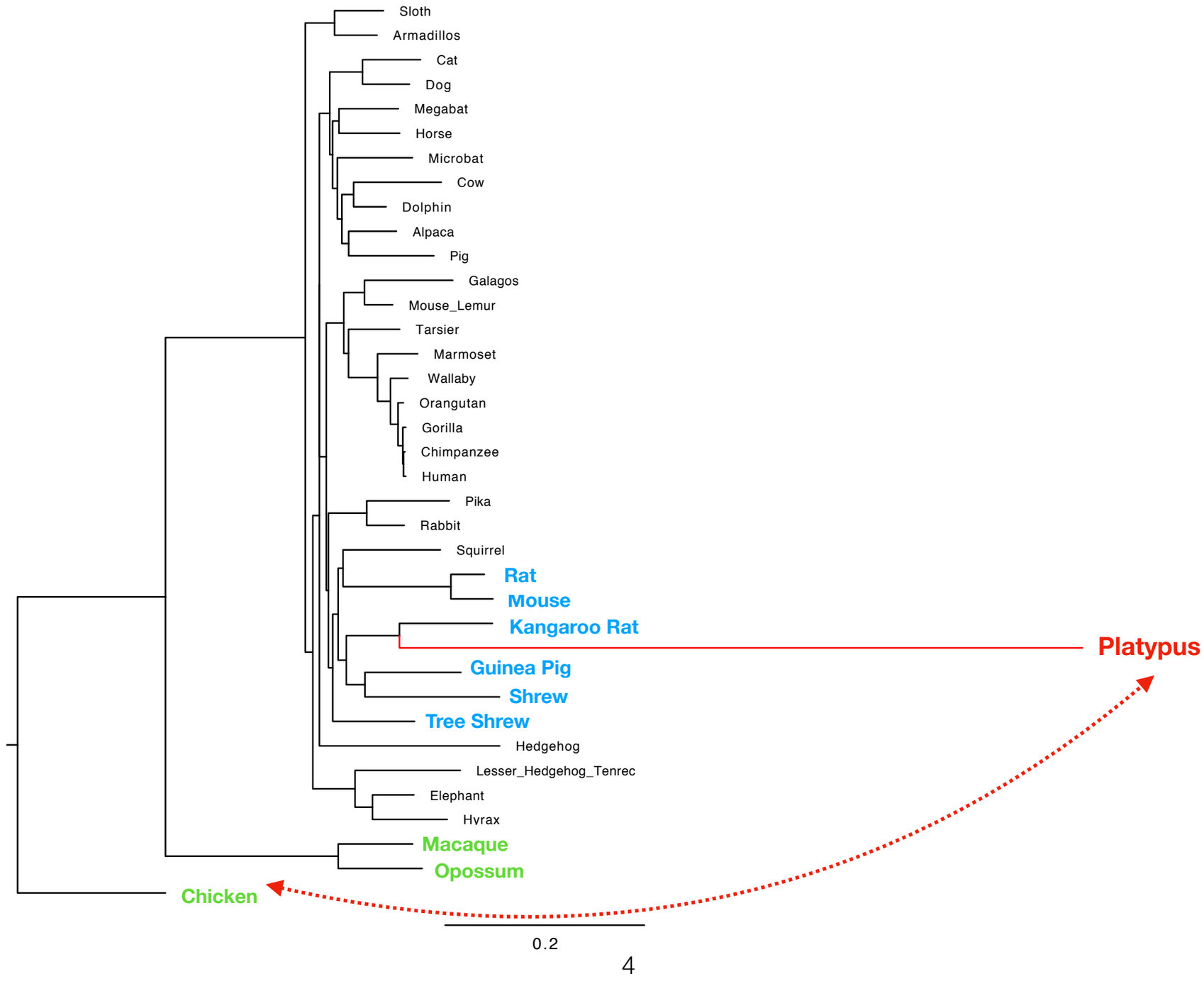
Observations

- Sequence data often include various sources of error
  - contamination
  - mistaken orthology
  - misalignment
- Erroneous sequences can appear as unproportionally long branches in the gene trees



- Deep coalescent nodes also appear on long branches
- Detecting long branches can be helpful in screening for errors in gene trees

From Gatesy et. al. (2014)



A Gene tree from Mammalian dataset
Song et al, PNAS, 2012

5

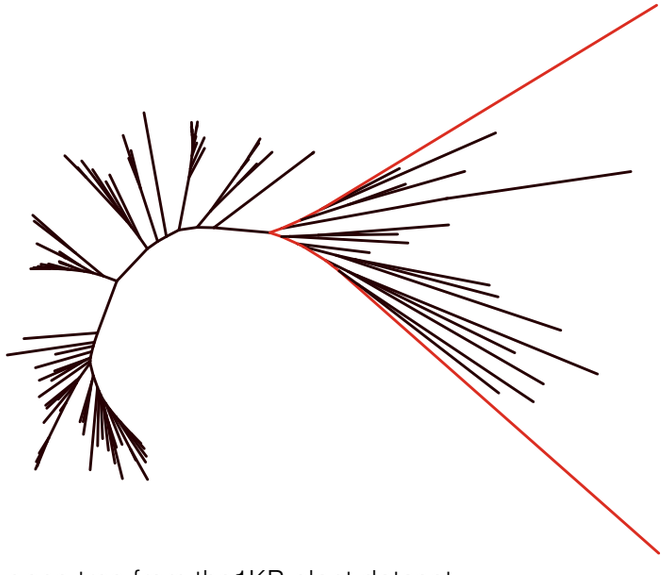
A Gene tree from 1kp Plants dataset

Wicket et al, PNAS, 2014

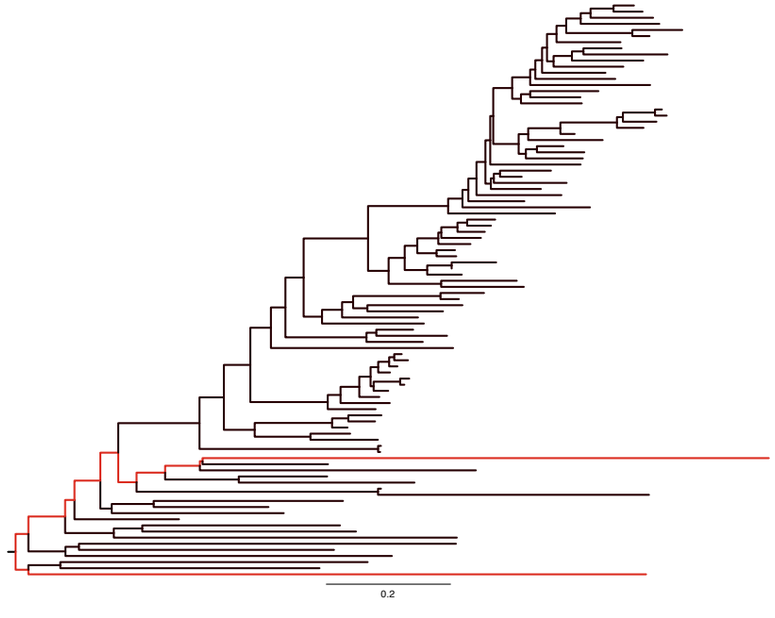
Q: How to detect long branches?

A: Remove leaves to maximally reduce the diameter

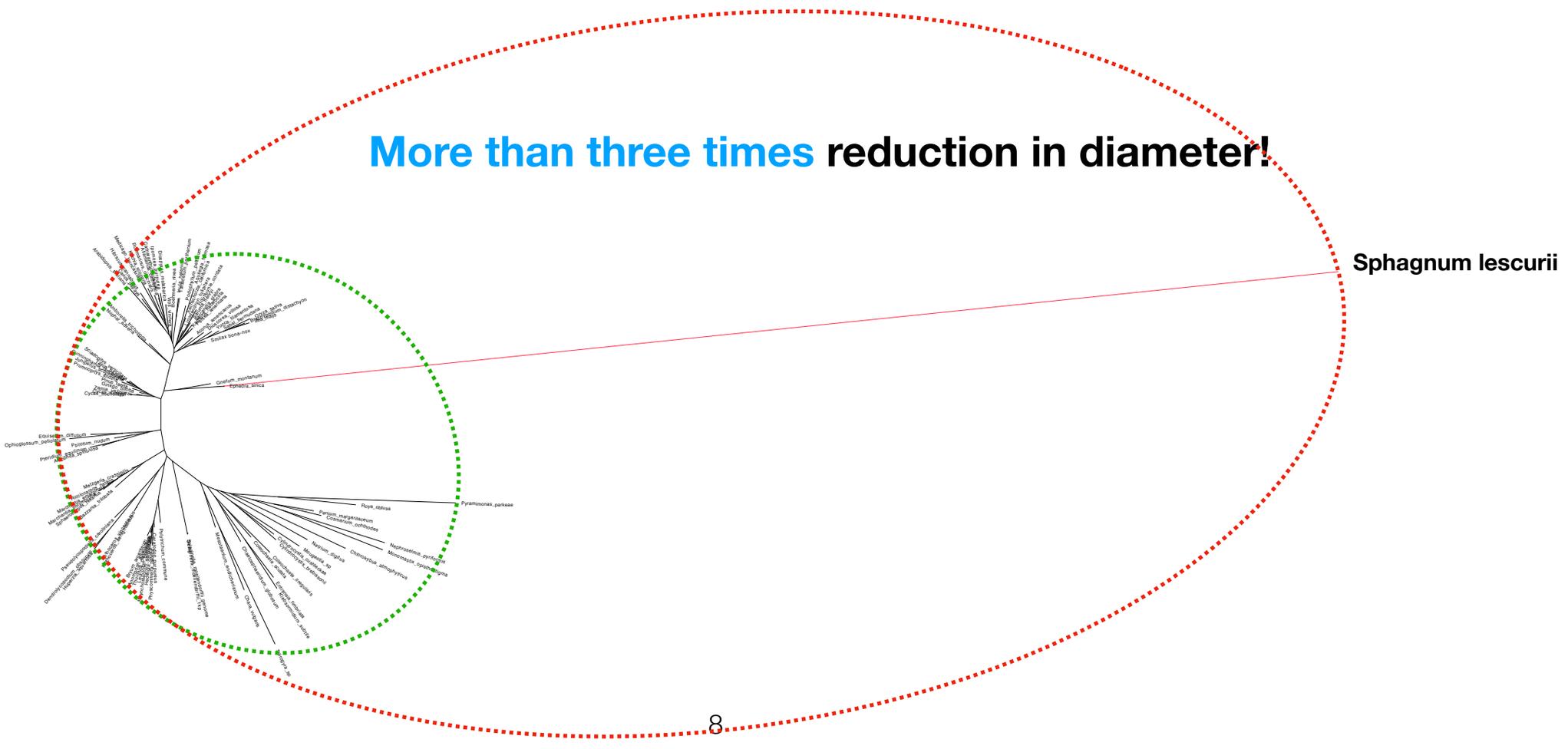
Diameter: The longest path between any two leaves



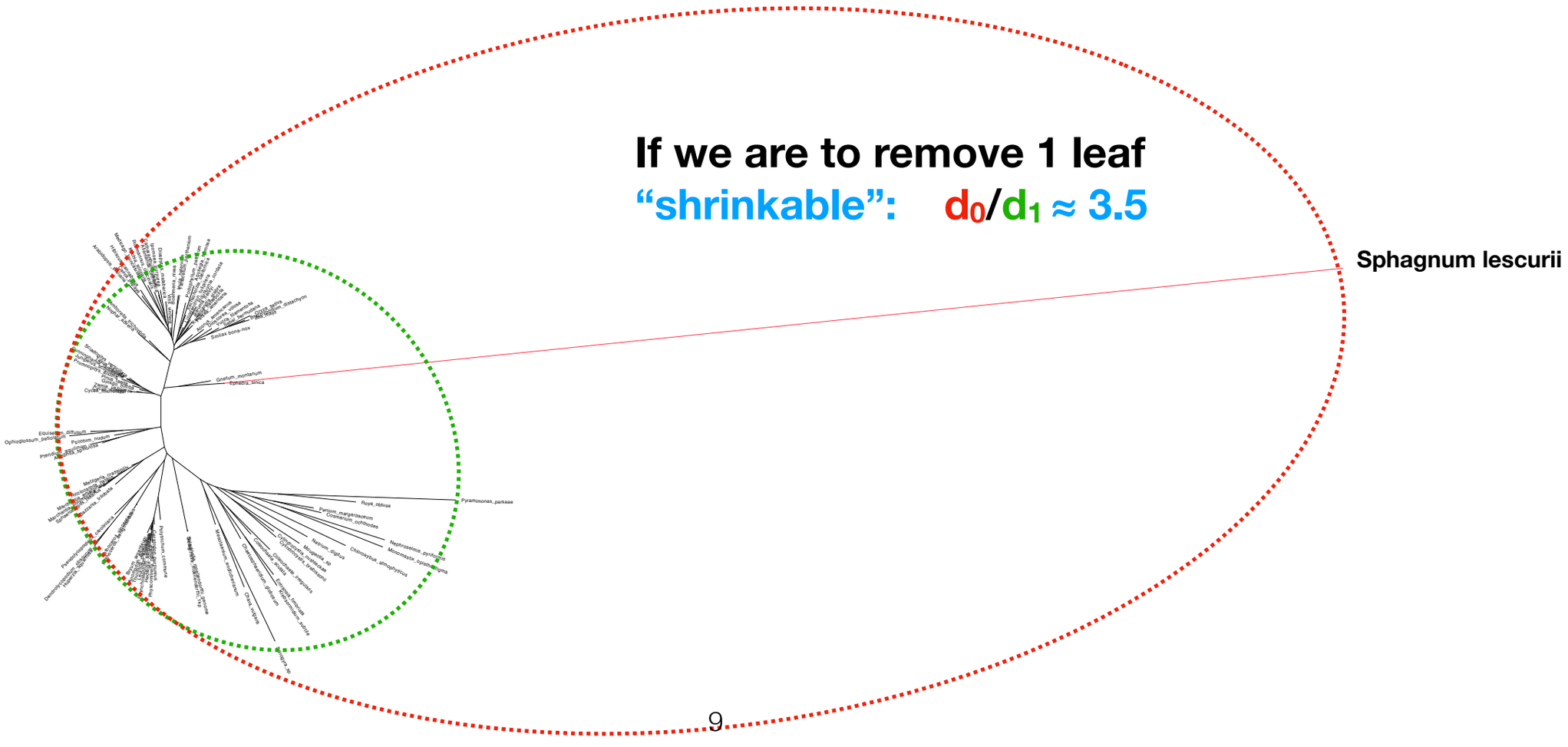
A gene tree from the1KP plant dataset
(Wicket *et al,* PNAS, 2014)



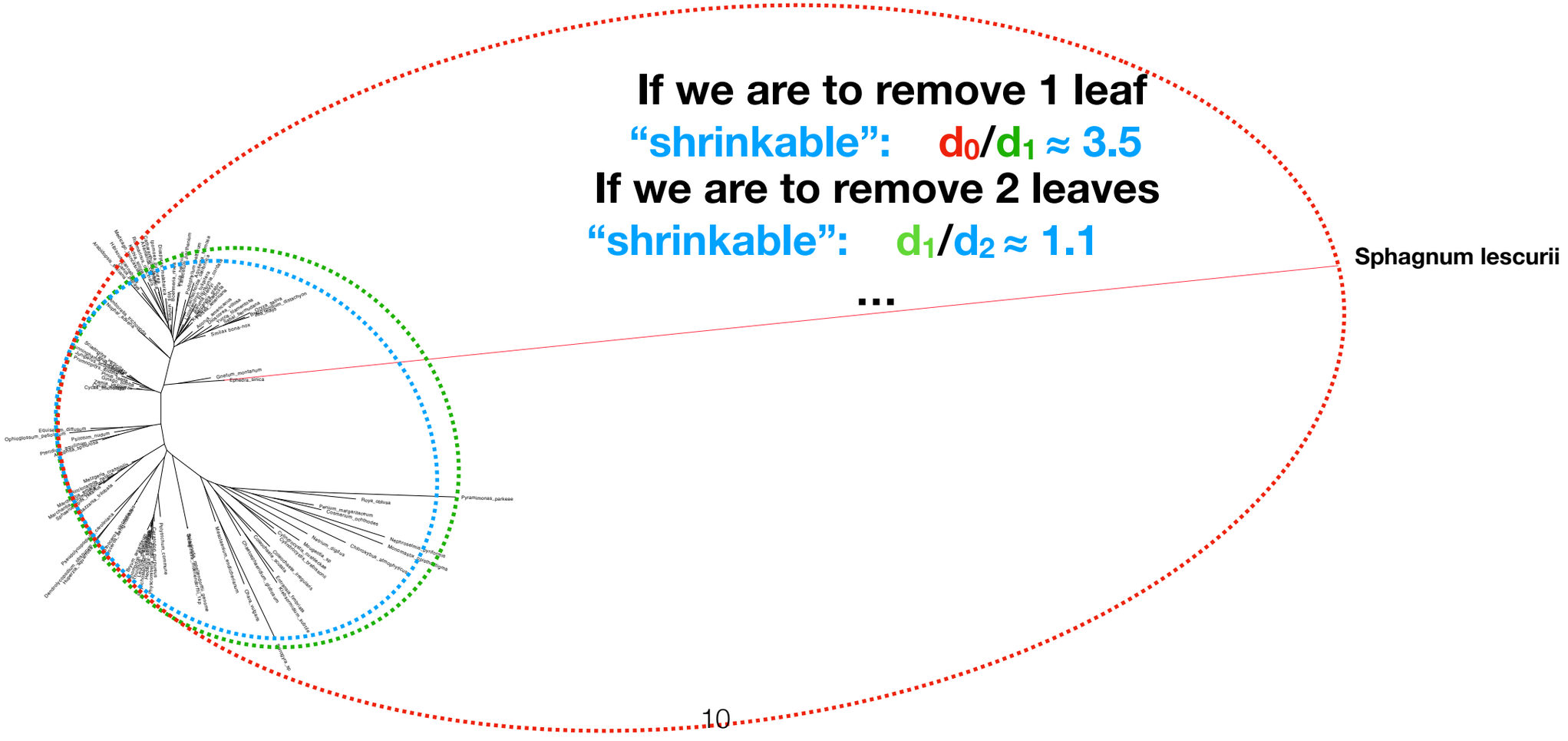
Diameter tracking



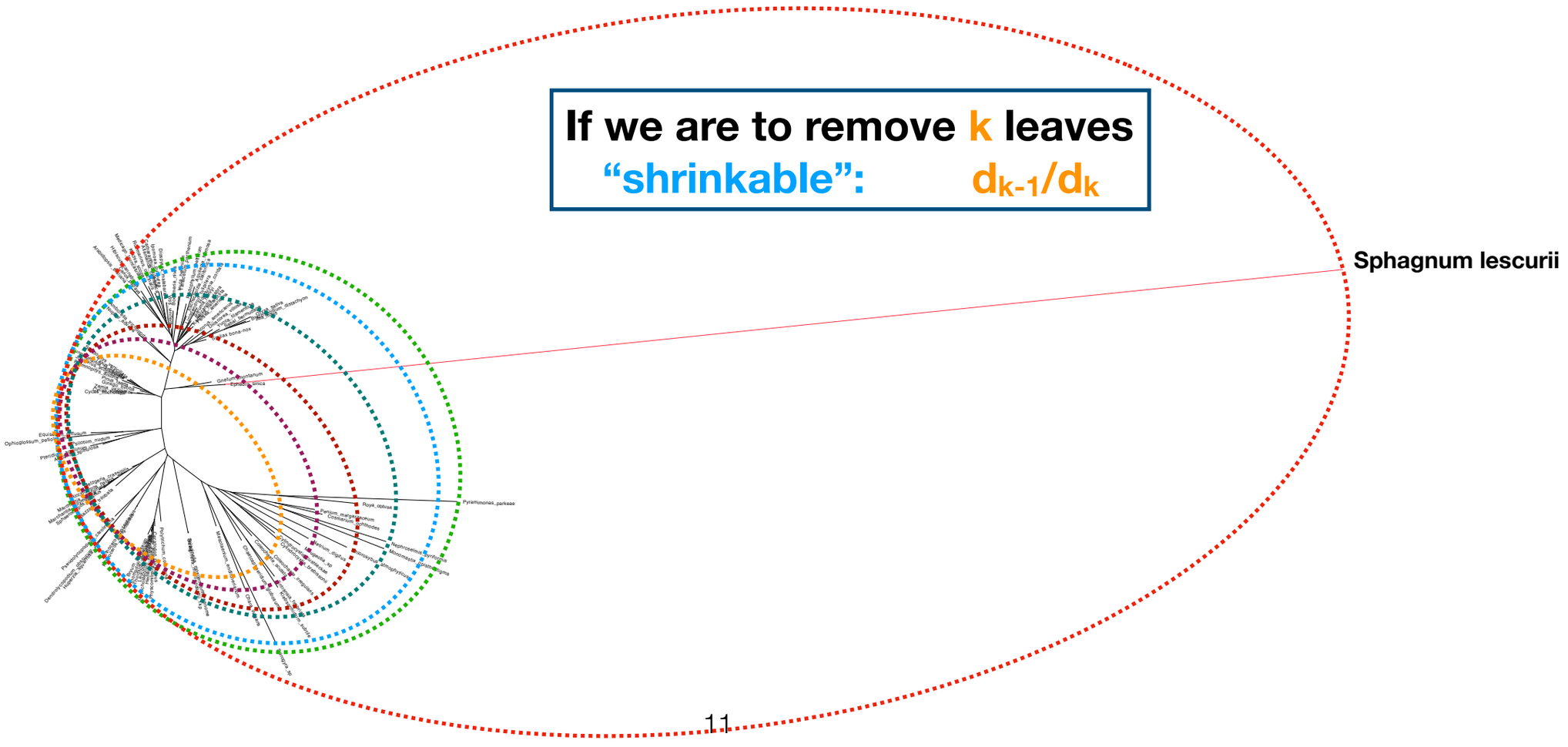
Diameter tracking



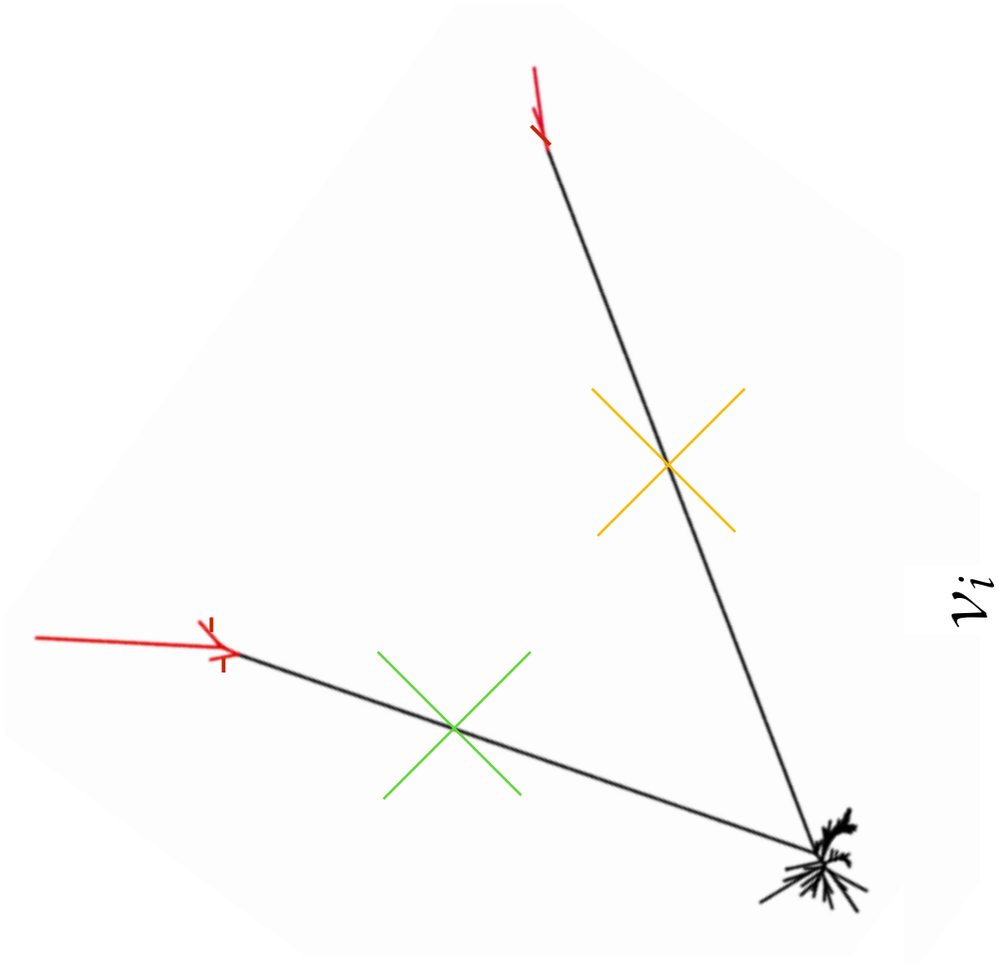
Diameter tracking

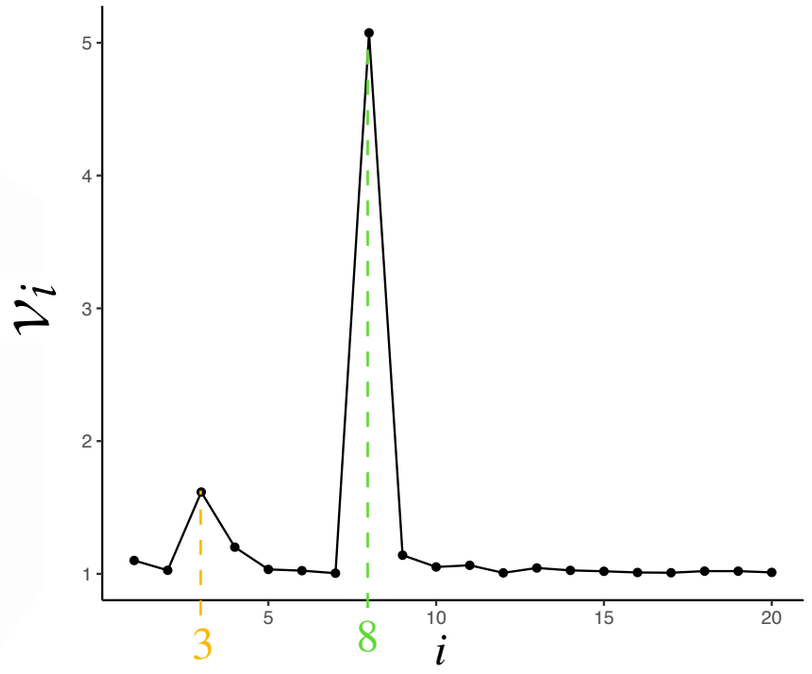


Diameter tracking



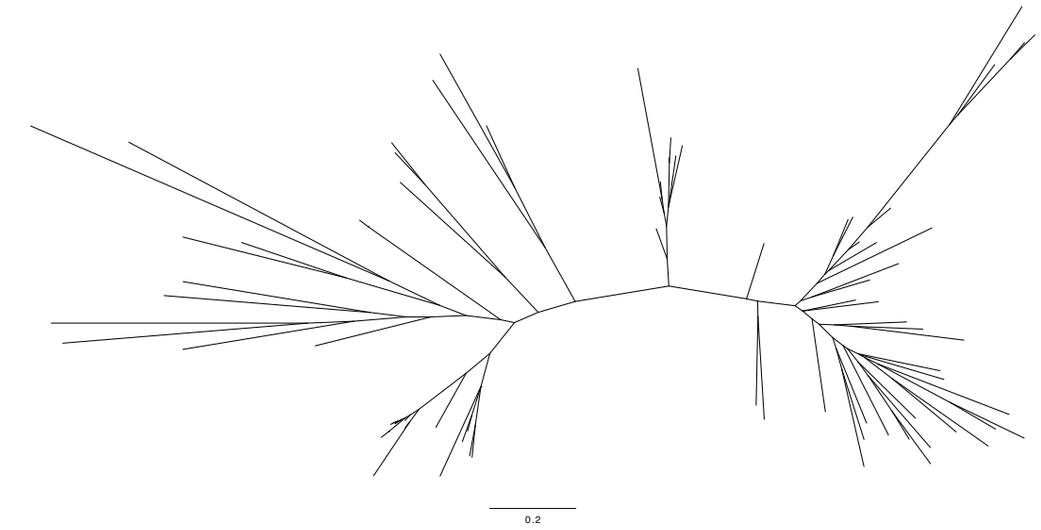
Diameter-shrinking plot





removal	shrinkable
$i = 1$	$\nu_1 = \frac{d_0}{d_1}$
$i = 2$	$\nu_2 = \frac{d_1}{d_2}$
$i = 3$	$\nu_3 = \frac{d_2}{d_3}$
$i = 4$	$\nu_4 = \frac{d_3}{d_4}$
…	…

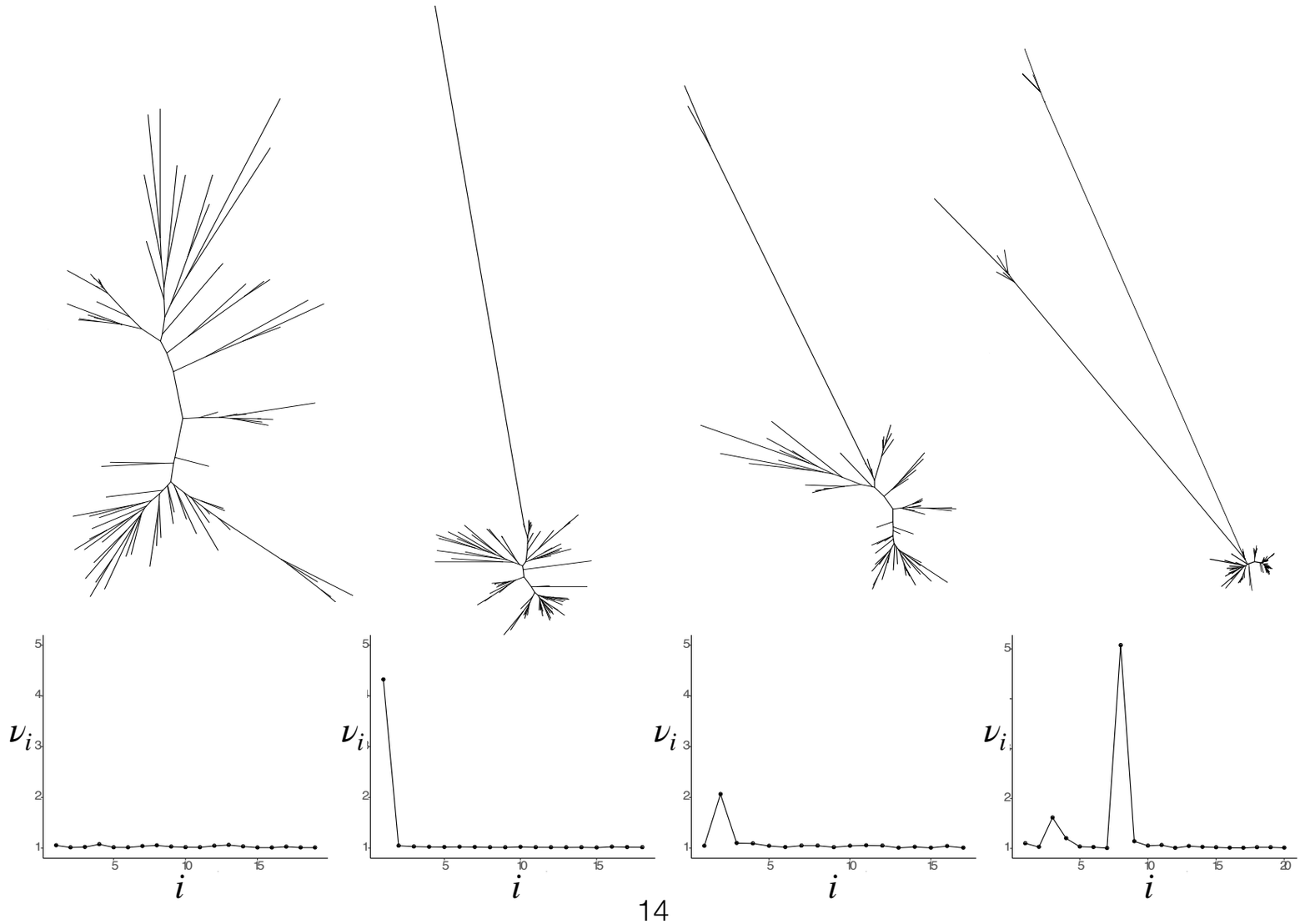
Diameter-shrinking plot



“flat” plot —> no outliers!

removal	shrinkable
$i = 1$	$\nu_1 = \frac{d_0}{d_1}$
$i = 2$	$\nu_2 = \frac{d_1}{d_2}$
$i = 3$	$\nu_3 = \frac{d_2}{d_3}$
$i = 4$	$\nu_4 = \frac{d_3}{d_4}$
…	…

Diameter-shrinking plot



Q: How to automate the process?

A: Use TreeShrink!

https://github.com/uym2/TreeShrink

Mai and Mirarab. BMC Genomics 2018

TreeShrink: Algorithm

Step 1: compute the sets of 1, 2, …, k leaves that could be removed to reduce the diameter maximally

Step 2: computer diameter-shrinking plots for 1…k

Step 3: Use a statistical test to detect outliers and suggest them for removal

TreeShrink: Installation


```
conda install -c smirarab treeshrink
```

TreeShrink: Installation


```
git clone https://github.com/uym2/TreeShrink.git

        python setup.py install [--user]
```

TreeShrink: Usage

```
run_treeshrink.py [-h] [-i INDIR] [-t TREE]
  [-a ALIGNMENT][-o OUTDIR][-q QUANTILES]
    [-m MODE][-c][-k K]
```

TreeShrink: Inputs

```
-i INDIR
          The parent input directory where the trees (and
alignments) can be found.

-t TREE
          The name of the input tree/trees. If the input
directory is specified (see -i option), each subdirectory
under it must contain a tree with this name. Otherwise, all
the trees can be included in this one file. Default:
input.tre

-a ALIGNMENT
          The name of the input alignment; can only be used
when the input directory is specified (see -i option). Each
subdirectory under it must contain an alignment with this
name. Default: input.fasta
```

TreeShrink: Outputs

```
-o OUTDIR
            Output directory. Default: the same as input directory (if
it is specified) or in the same directory with the input trees.
```

- The output directory will include
  - The removing list: the species removed from each input tree
  - The shrunk trees: the trees with the suggested species removed.
  - The filtered alignments: the alignments (if provided) with suggested species removed.

TreeShrink: Example

```
run_treeshrink.py -t test_data/mm10.trees
-o test_data/mm10_treeshrink
```

- Inside the generated folder `test_data/mm10_treeshrink/`
  - the shrunk trees `mm10_shrunk_0.05.trees`
  - the removing set `mm10_shrunk_RS_0.05.txt`

TreeShrink: include alignment

- Alignments can optionally be included to be filtered with the trees

- The alignments do not have any impact on outlier detection. They will be filtered based on the results of the filter applied to the trees.

- To include alignments, use -a together with -i

TreeShrink: Example

```
> ls allgenes/*

allgenes/4048:
tree.nwk
alignment.fasta

allgenes/4103:
tree.nwk
alignment.fasta

allgenes/4218:
tree.nwk
alignment.fasta

allgenes/4234:
tree.nwk
alignment.fasta
```

TreeShrink: Example

```
run_treeshrink.py -i allgenes -t tree.nwk -a
alignment.fasta
```

```
allgenes/4048:
tree.nwk
tree_shrunk_0.05.nwk
tree_shrunk_RS_0.05.txt
alignment.fasta
alignment_shrunk0.05.fasta

allgenes/4103:
tree.nwk
tree_shrunk_0.05.nwk
tree_shrunk_RS_0.05.txt
alignment.fasta
alignment_shrunk0.05.fasta
```

```
allgenes/4218:
tree.nwk
tree_shrunk_0.05.nwk
tree_shrunk_RS_0.05.txt
alignment.fasta
alignment_shrunk0.05.fasta

allgenes/4234:
tree.nwk
tree_shrunk_0.05.nwk
tree_shrunk_RS_0.05.txt
alignment.fasta
alignment_shrunk0.05.fasta
```

TreeShrink: -m option

- TreeShrink includes three modes
  - per-gene
  - all-genes
  - per-species
- By default, TreeShrink automatically selects an appropriate mode (usually per-species is chosen)
- Use -m to manually change the mode

TreeShrink: -q and -b

- To control the sensitivity of TreeShrink, use **-q** and **-b**

```
-q QUANTILES
              The false-tolerance threshold. Multiple thresholds can be
specified. Default: 0.05

-b MINIMPACT
              To be used with per-species mode. The minimum impact
(percent) on the diameter on which the species could be removed. As
such, TreeShrink never removes the species if their impact on diameter
is less than MINIPACT%. Default: 5
```

TreeShrink: -k and -s

- Use -k and -s to set the size of the diameter-shrinking plot (x-axis)
- In the per-gene mode, this number is the maximum number of species that *could be removed* per tree

```
-k K
     The size of the diameter-shrinking plot; i.e. maximum number of
leaves that can be removed. Default: auto-select based on the data

-s KSCALING
     If -k is not given, we use k=min(n/a,b*sqrt(n)) by default; using
this option, you can set the a,b constants; Default: '5,2'
```

TreeShrink: Example

```
run_treeshrink.py  -t test_data/mm50.trees
-q "0.05 0.10” -b 5 -k 5 -m per-species
—o test_data/mm50_treeshrink_multi
```

- Generate folder `test_data/mm50_treeshrink_multi/` which contains two sets of outputs
  - at α = 0.05
    - `mm50_shrunk_0.05.trees` and `mm50_shrunk_RS_0.05.txt`
  - at α = 0.10
    - `mm50_shrunk_0.1.trees` and `mm50_shrunk_RS_0.1.txt`

TreeShrink: Logging

```
Launching TREESHRINK version 1.3.3

TREESHRINK was called as follow

run_treeshrink.py -t test_data/mm50.trees -m per-species –q 0.05 0.10 -b 5 -k 5
-o test_data/mm50_treeshrink_multi

Solving k-shrink with k = 5
Solving k-shrink with k = 5
Solving k-shrink with k = 5
…

TreeShrink will run in 'Per-species' mode ...
CAV:
     will be cut in 2 trees where its impact is above 1.212920 for quantile 0.05
CAV:
     will be cut in 5 trees where its impact is above 1.119273 for quantile 0.10
DAS:
     will be cut in 1 trees where its impact is above 1.050000 for quantile 0.05
DAS:
     will be cut in 1 trees where its impact is above 1.050000 for quantile 0.10
…
```

References

Publication

Mai, Uyen, and Siavash Mirarab. “TreeShrink: Fast and Accurate Detection of Outlier Long Branches in Collections of Phylogenetic Trees.” BMC Genomics 19, no. S5 (2018): 272. doi:10.1186/s12864-018-4620-2.

TreeShrink Software

https://github.com/uym2/treeshrink

Contact

Uyen Mai

umai@eng.ucsd.edu