Reading and Making Phylogenetic Trees in R

Simon Joly

BIO 6008 - Fall 2015

Contents

Normally, I would not recommand performing phylogenetic analyses in R. But if you want a simple tree to work with, it is possible to build it in R.

Importing sequence data

The best is to import sequence data that is aligned and either in fasta or phylip. These are the prefered format in `ape`. To import sequences in fasta format, use the following command.

```
require(ape)
rbcl <- read.FASTA("./data/rbcL.fasta")
```

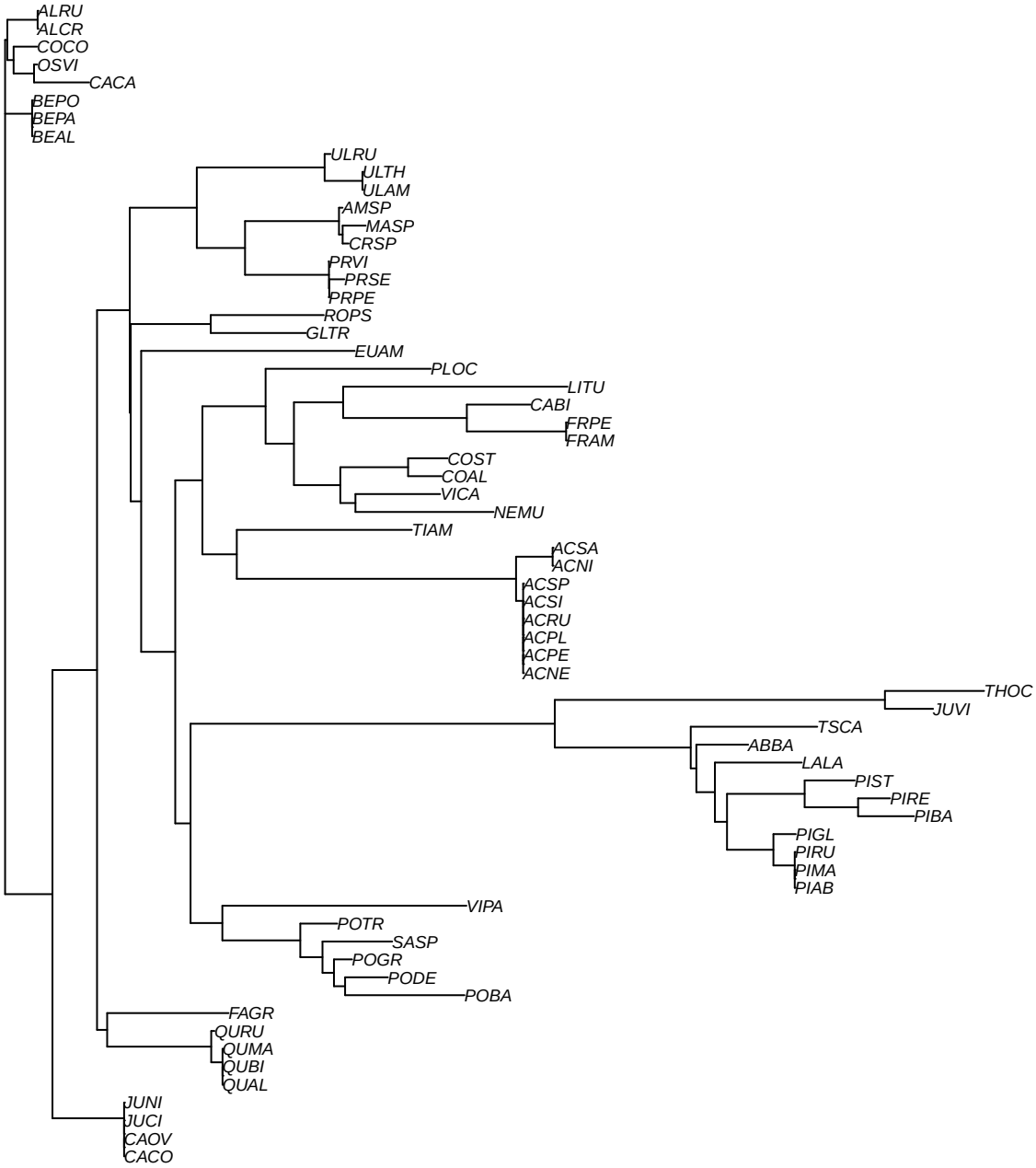
To read sequences in phylip format, use instead this command.

```
require(ape)
rbcl <- read.dna("./data/rbcL.phy")
```

Tree reconstruction

Here, we will use the neighbor-joinning algorithm to reconstruct a phylogeny. This is not the best algorithm, but is it not bad and it is fast. We need a distance matrix to use this method. Here, we get a distance matrix using the K80 nucleotide substitution model. Again, this is just for the demonstration. Normally, you would have to select the best substitution model for your data.

```
rbcl.k80 <- dist.dna(rbcl,model="K80")
rbcl.tree <- nj(rbcl.k80)
plot(rbcl.tree,cex=0.6, no.margin=TRUE)
```



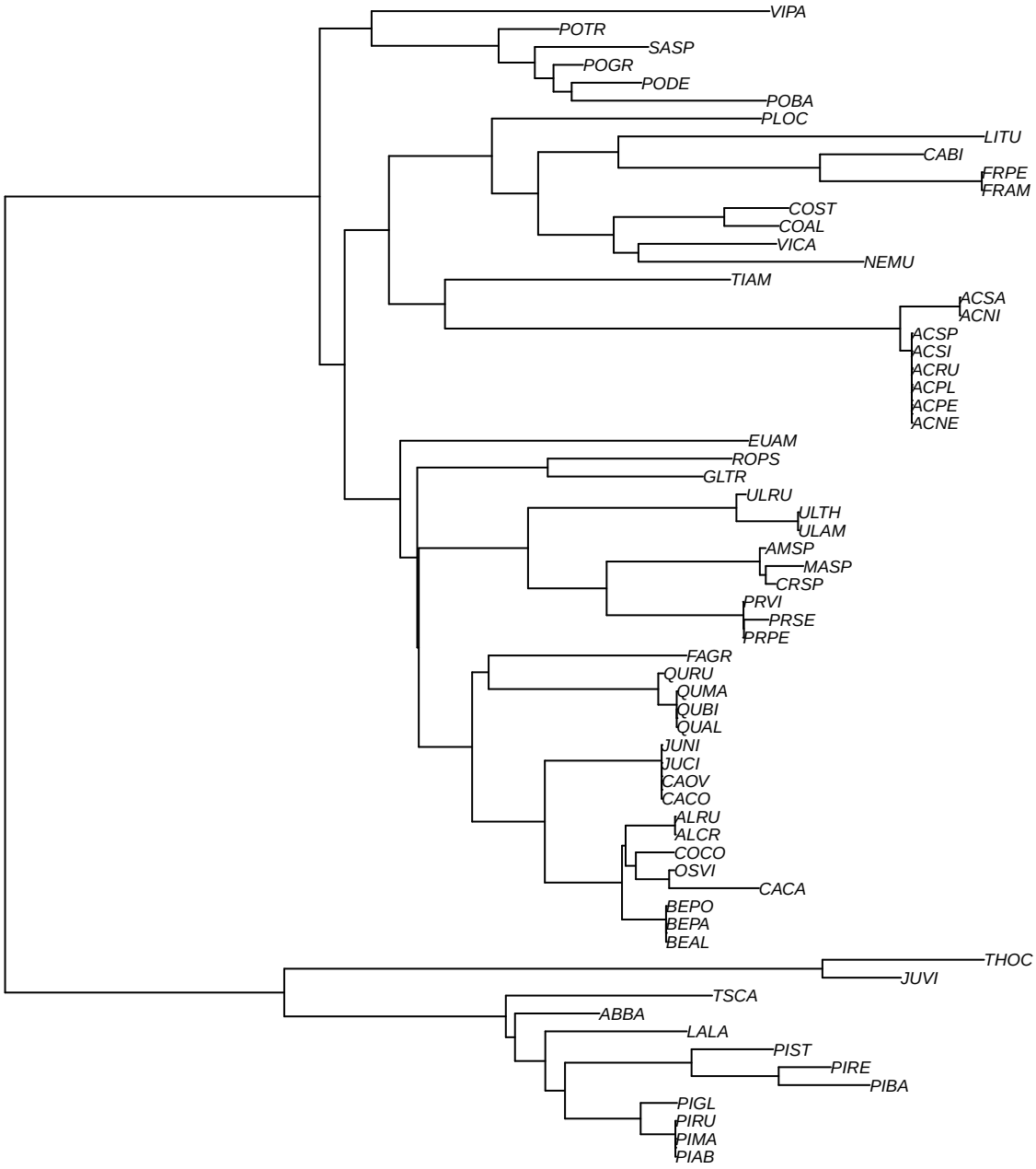
Root tree

You may need to root the tree with the most ancestral sequence. You can do this using the function `root`

```
rbcl.tree.rooted <- root(rbcl.tree,outgroup="JUVI")
plot(rbcl.tree.rooted,cex=0.6, no.margin=TRUE)
```

If you do not have an outgroup, you could use midpoint rooting, which places the root in the middle of the longest path between two species.

```
require(phangorn)
rbcl.tree.rooted <- midpoint(rbcl.tree)
plot(rbcl.tree.rooted,cex=0.6, no.margin=TRUE)
```



Chronogram

For most phylogenetic comparative methods, it makes sense to use a chronogram, that is a ree with the species that are all equidistant from the root. We also call such trees ultrametric trees. Chronogram can be reconstructed using Bayesian approaches (BEAST, MrBayes), but you can also obtain them using the chronos function in `ape`. Note, however, that the trees obtained with this function might not reflect divergent times, so be careful. But it will be useful for the tutorials of the course.

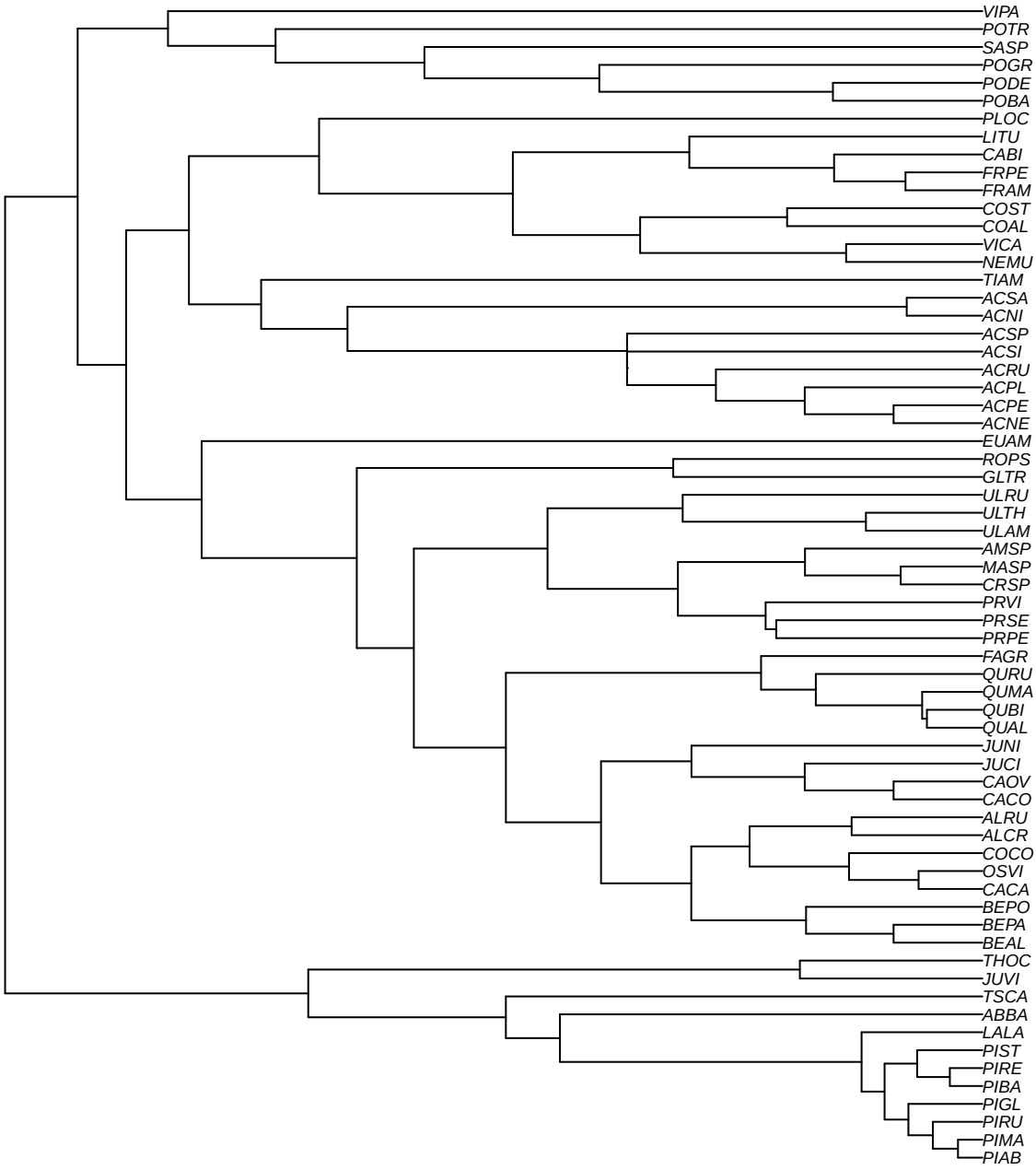
```
# For unknown reasons, the neighbor-joinning method resulted in
# negative branch lengths. Let's give them a length of 0.
rbcl.tree.rooted$edge.length[rbcl.tree.rooted$edge.length<0] <- 0
rbcl.chrono <- chronos(rbcl.tree.rooted, model="relaxed")
```

```
##
## Setting initial dates...
## Fitting in progress... get a first set of estimates
##          Penalised log-lik = -9.521653
## Optimising rates... dates... -9.521653
## Optimising rates... dates... -9.494325
## Optimising rates... dates... -9.486704
## Optimising rates... dates... -9.398799
##
## Done.
```

```
#  Check that it is ultrametric
is.ultrametric(rbcl.chrono)
```

```
## [1] TRUE
```

```
plot(rbcl.chrono, cex=0.6, no.margin=TRUE)
```



By looking at the tree, you can see that the function is problematic. For instance, all the Acer (maple trees; accessions starting with ACxx) are identical for the gene *rbcL*, but after using Chronos they appear to be quite different...

If you want to use your chronogram for other analyses, you might have to convert it into a `phylo` object. The simplest way I found of doing so is the following:

```
# Convert chronogram to phylo
rbcl.chrono <- read.tree(text=write.tree(rbcl.chrono))
```

Import tree

Tree format

Ideally, you should have built your tree in an appropriate software and load it in R. The pacakge `ape` accepts two trees formats: nexus and newick.

The newick format

The newick format is the strict parenthetic format. For example, here is a newick tree with 4 species:

```
(((species1:4.2,species2:4.2):3.1,species3:7.3):6.3,species4:13.5);
```

To read a tree in newick format, you need to use to function `read.tree` from `ape`

```
tree <- read.tree("./data/rbcl.newick")
```

The nexus format

The nexus format is slightly more complex and looks like this:

```
#NEXUS
begin taxa;
  dimensions ntax=4;
    taxlabels
    species1
    species2
    species3
    species4
;
end;

begin trees;
  tree1 tree_name = [&U] (((species1:4.2,species2:4.2):3.1,species3:7.3):6.3,species4:13.5);
end;
```

To read it, you will need the `read.nexus` function.

```
tree <- read.nexus("./data/rbcl.tre")
```

Note that Bayesian trees can be read as well. See the tutorial of the first lecture.