Ancestral state reconstruction

Simon Joly

BIO 6008 - Fall 2015

Contents

Quantitative states

There are two main options for reconstructing the ancestral states of continuous variables: reconstructions based on maximum likelihood (ML) and reconstructions based on phylogenetic independent contrasts (pic). However, the pic apprach is less interesting because it only considers the parameter values of the descendant nodes (and not of the whole tree) when estimating the estimates at a node. For this reason, we will focus here on the ML approach.

Prepare seed plant data

Throughout this tutorial, we will use the seed plant phylogeny and trait data from Paquette et al. (2015). Let's load it and prepare it.

```
# Load ape
require(ape)
# Import datasets
seedplantstree <- read.nexus("./data/seedplants.tre")
seedplantsdata <- read.csv2("./data/seedplants.csv")
# Remove species for which we don't have complete data
seedplantsdata <- na.omit(seedplantsdata)
# Remove species in the tree that are not in the data matrix
species.to.exclude <- seedplantstree$tip.label[!(seedplantstree$tip.label %in%
```

```
                                                    seedplantsdata$Code)]
seedplantstree <- drop.tip(seedplantstree,species.to.exclude)
rm(species.to.exclude)
# Name the rows of the data.frame with the species codes used as tree labels
rownames(seedplantsdata) <- seedplantsdata$Code
seedplantsdata <- seedplantsdata[,-1]
# Order the data in the same order as the tip.label of the tree. In the present
# example, this was already the case.
seedplantsdata <- seedplantsdata[seedplantstree$tip.label,]
# Create a factor for a categorical variable
height <- factor(seedplantsdata$height)
names(height) <- rownames(seedplantsdata)
# Create a vector for a continuous character
maxH <- seedplantsdata$maxH
names(maxH) <- rownames(seedplantsdata)
```

Maximum likelihood

It is possible to reconstruct ancestral states by maximum likelihood using the Brownian Motion model. This is the method of Schluter et al. (1997), implemented in the `ace` function of the `ape` package.

```
# Ancestral state reconstruction
MLreconstruction <- ace(maxH, seedplantstree, type="continuous",
                        model="BM", method="ML")
# The maximum likelihood estimates:
MLreconstruction$ace
```

```
##        58        59        60        61        62        63        64
## 21.205204 21.698583 21.211068 20.363586 24.066581 26.821218 21.330449
##        65        66        67        68        69        70        71
## 21.122281 21.254648 21.558789 21.190270 14.738688 11.237791 14.276062
##        72        73        74        75        76        77        78
##  6.908920 18.537740 24.117904 25.841500 24.992299 26.993450 22.682126
##        79        80        81        82        83        84        85
## 25.310547 25.987239 19.303439 21.599492 27.129715 29.905906 17.807506
##        86        87        88        89        90        91        92
##  9.912892  9.359468  8.821772 16.560892 10.943410 18.907020 23.154049
##        93        94        95        96        97        98        99
## 23.252838 22.746764 22.445718 20.038876 20.348376 19.336762 21.242429
##       100       101       102       103       104       105       106
## 31.799269 13.820609 29.470598 21.082006 21.936068 22.256714 22.480315
##       107       108       109       110       111       112       113
## 25.441178 26.538339 31.435089 23.762154 22.711235 24.012507 17.531736
```

```
# The 95% confidence interval around the estimates
MLreconstruction$CI95
```

```
##          [,1]     [,2]
## 58  -4.338803 46.74921
## 59   7.250033 36.14713
## 60   9.493639 32.92850
```

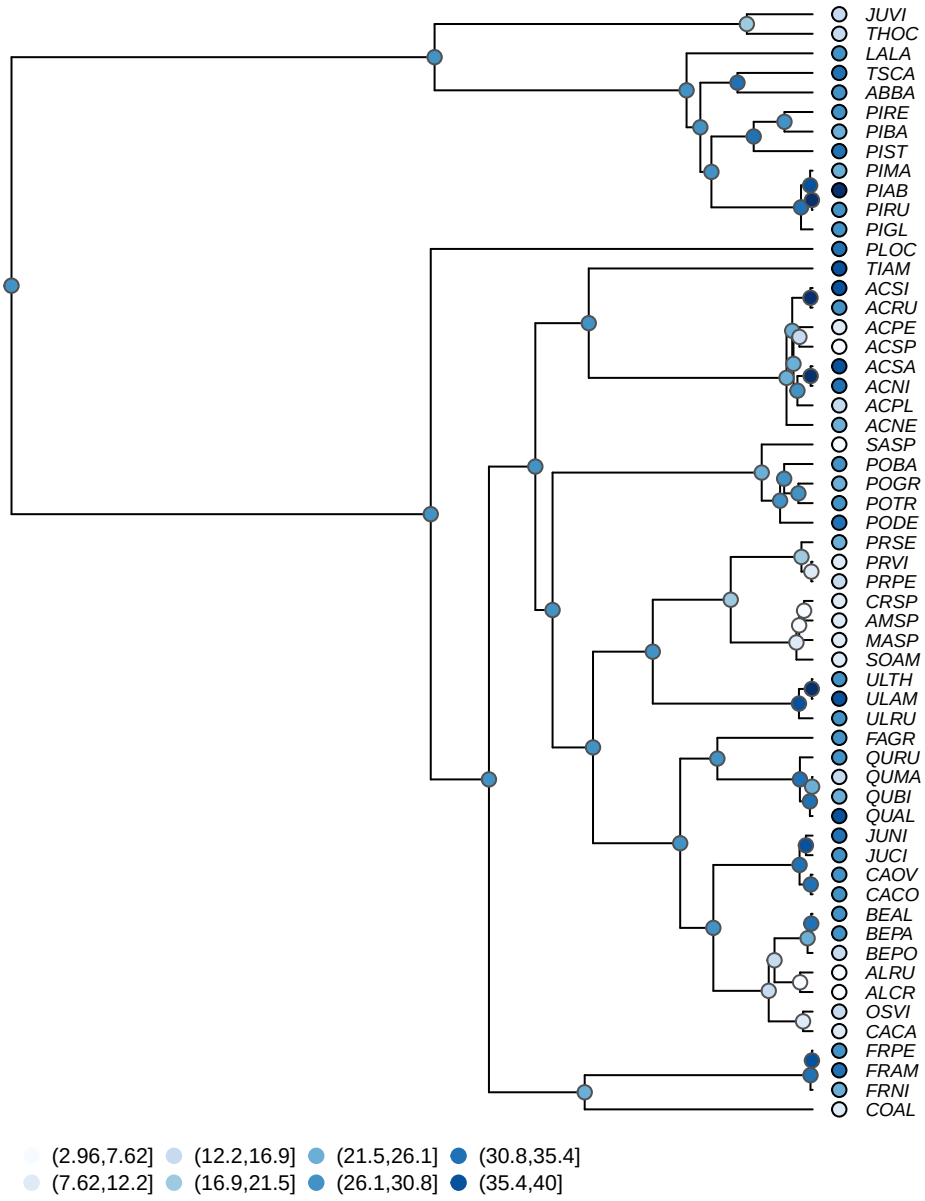
```
## 61   7.521974 33.20520
## 62  22.528594 25.60457
## 63  25.936761 27.70568
## 64  11.185157 31.47574
## 65  11.128843 31.11572
## 66  11.463637 31.04566
## 67  12.994377 30.12320
## 68  13.132091 29.24845
## 69   9.262176 20.21520
## 70   7.989336 14.48625
## 71   9.090678 19.46145
## 72   3.282757 10.53508
## 73  16.217327 20.85815
## 74  22.860484 25.37532
## 75  22.319545 29.36346
## 76  23.510192 26.47441
## 77  24.389142 29.59776
## 78  14.222274 31.14198
## 79  21.609214 29.01188
## 80  24.263412 27.71107
## 81  18.520384 20.08649
## 82  11.544743 31.65424
## 83  23.174387 31.08504
## 84  28.895525 30.91629
## 85   9.423900 26.19111
## 86   6.386700 13.43908
## 87   6.209204 12.50973
## 88   6.056742 11.58680
## 89  13.032128 20.08966
## 90   9.708616 12.17820
## 91  12.465125 25.34891
## 92  18.456410 27.85169
## 93  18.811671 27.69401
## 94  19.061524 26.43200
## 95  10.991798 33.89964
## 96  15.799580 24.27817
## 97  17.082100 23.61465
## 98  16.211531 22.46199
## 99  18.033807 24.45105
## 100 30.288044 33.31049
## 101 10.624266 17.01695
## 102 27.890794 31.05040
## 103  3.187733 38.97628
## 104 13.369944 30.50219
## 105 14.688355 29.82507
## 106 15.017139 29.94349
## 107 21.904802 28.97755
## 108 24.916624 28.16006
## 109 30.520867 32.34931
## 110 16.982459 30.54185
## 111 17.448558 27.97391
## 112 16.315068 31.70995
## 113  8.916805 26.14667
```

Once you have inferred the ancestral states, you can visualize them on the tree.

```
# Breakdown continuous trait in categories
maxH.cat <- cut(maxH,breaks=9,labels=FALSE)
# Same thing for internal nodes
maxH.nodes.cat <- cut(MLreconstruction$ace,breaks=9,labels=FALSE)
# Color palette
require(RColorBrewer)
```

```
## Loading required package: RColorBrewer
```

```
ColorPalette <- brewer.pal(n = 9, name = "Blues")
# Plotting
plot(seedplantstree, type="p", use.edge.length = TRUE, label.offset=0.01,cex=0.6)
tiplabels(pch=21,bg=ColorPalette[maxH.cat],col="black",cex=1,adj=0.505)
nodelabels(pch=21,bg=ColorPalette[maxH.nodes.cat],cex=1,col="gray32",)
op<-par(xpd=TRUE)
legend(0,0,legend=levels(cut(maxH,breaks=8)),
       col=ColorPalette,pch=20,bty="n",cex=0.7,pt.cex=1.5,ncol=4)
```

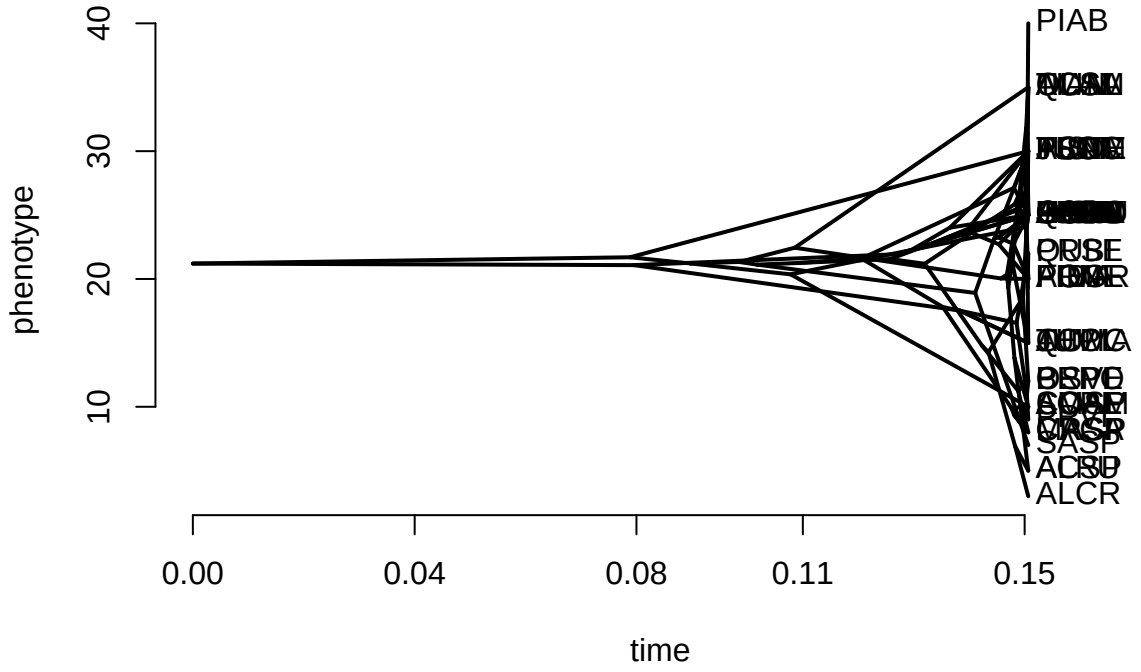



```
par(op) #reset graphical parameters to defaults
```

Warning

It is important to note that the reconstruction will be good **if** the model is good. Unfortunately, this is difficult to know. Also, because the reconstruction depends on the observed states and because of the existing models, it will never be possible to obtain an ancestral state at a node that is more extreme than the decesndant nodes. This is even more evident when looking at a traitgram.

```
require(phytools)
phenogram(seedplantstree,c(maxH,MLreconstruction$ace))
```



For these reasons, it is important to take great care when interpreting these results.

Qualitative states

It is also possible to reconstruct ancestral character states for qualitative (discrete) characters. There are a lot more options for these analyses.

Joint versus marginal state reconstruction

There are two types of ancestral state reconstruction possible with qualitative (discrete) traits: marginal and joint. The joint estimation will find the best combination of ancestral states over all nodes of the tree. In contrast, the marginal estimation will maximize the probability of the states at a node, integrating over all possible states at all other nodes (that is, considering all the possible states at those other nodes). In some instances, the two approaches can give very different results.

Prior probabilities at the root

In any reconstructions, we need to give a prior on the probabilities of the different states at the root. This can be set 1) at equal probabilities for all states (a flat prior), 2) using the stationary distribution of the model, or 3) using the empirical frequency amongst the species sampled. This choice can have important repercussions. This is especially true because there is little useful data to infer the states at the root, so the prior probability dominates the posterior. As we will see below with examples, the root state probabilities is often very close to the prior probabilities. For this reason, it is better not to interpret the results obtained at the root of a phylogeny (Gascuel and Steel, 2014).

Test different evolutionary models

To reconstruct the ancestral state at the nodes of a tree with makimum likelihood, we need an evolutionary model (for a good introduction to likelihood models, see Baum and Smith, 2012). This model will dictate

how frequently character states change between them. The models can be anything between the simplest model where all rates are different to the most complex where all transition rates are different. These models are generally represented by a transition matrix that quantifies the probability of change from one state to the other in any infiniesimal amount of time. For instance, the following matrix represent the simplest model for three characters states (1, 2, 3), the equal rates (ER) model:

	1	2	3
1	-	q	q
2	q	-	q
3	q	q	-

The q represents the instantaneous rate between these states in an infinitesimal amount of time. You can notice that with this ER model, all rates are the same. The main diagonal elements are constrained to be equal to minus the sum of the other elements in the row.

In contrast, the following table represent the All Rates Different (ARD) model:

	1	2	3
1	-	q_{12}	q_{13}
2	q_{21}	-	q_{23}
3	q_{31}	q_{32}	-

In the preceeding table, q_{12} represents the instataneous transition rate from character 1 to character 2. Note that this model is not symmetric. That is $q_{12} \neq q_{21}$.

But which model should we choose? A good approach is to select the model that gives the best fit to the data (it maximizes it's likelihood). We will make a test with the seed plant data on plant height. We'll compare the equal rates (ER) and the all rates different (ARD) model. Also, we will also test a custom model in which there are two rates: one rate between neighboring categories (small <-> medium; medium <-> large) and another one for the other (small <-> large). This custom model is symmetric because the rate is the same in both directions.

We will compare the models using the AICc, which is the AIC corrected for small sample sizes. We will use the `fitDiscrete` function of the `geiger` package for this, although it could have been done with `ape`.

```
library(geiger)
# Create a table to store de results
results.anc <- data.frame(model=c("ER","ARD","custom"),
                          lnL=numeric(3),AICc=numeric(3),params=numeric(3))
# Fit ER
ER_fit <- fitDiscrete(seedplantstree,height,model="ER")
# Store the results
results.anc[1,-1]<- c(lnL=ER_fit$opt$lnL,AICc=ER_fit$opt$aicc,ER_fit$opt$k)
# Fit ARD
ARD_fit <- fitDiscrete(seedplantstree,height,model="ARD")
# Store the results
results.anc[2,-1]<- c(lnL=ARD_fit$opt$lnL,AICc=ARD_fit$opt$aicc,ARD_fit$opt$k)
# Fit custom model
custom_fit <- fitDiscrete(seedplantstree,height,
                          model=matrix(c(0, 1, 1, 1, 0, 2, 1, 2, 0), 3))
# Store the results
results.anc[3,-1]<- c(lnL=custom_fit$opt$lnL,AICc=custom_fit$opt$aicc,custom_fit$opt$k)
```

```
# Order the results by AICc
results.anc <- results.anc[order(results.anc$AICc),]
results.anc
```

```
##    model       lnL      AICc params
## 2    ARD -39.31349  92.30697      6
## 3 custom -45.25178  94.72579      2
## 1     ER -49.19547 100.46366      1
```

You can see that the ARD model provides the best fit to the data and that the ER model the worst. You can have a look at the ARD model to see the estimated rates:

```
ARD_fit
```

```
## GEIGER-fitted comparative model of discrete data
##  fitted Q matrix:
##                medium         small          tall
##     medium -173.90672  3.306114e+01  1.408456e+02
##     small   175.37196 -1.753720e+02  1.766867e-27
##     tall     60.20071  4.341428e-20 -6.020071e+01
##
##  model summary:
##  log-likelihood = -39.313486
##  AIC = 90.626972
##  AICc = 92.306972
##  free parameters = 6
##
## Convergence diagnostics:
##  optimization iterations = 100
##  failed iterations = 0
##  frequency of best fit = 0.14
##
##  object summary:
##  'lik' -- likelihood function
##  'bnd' -- bounds for likelihood search
##  'res' -- optimization iteration summary
##  'opt' -- maximum likelihood parameter estimates
```

By looking at the Q matrix (the matrix of instantaneous change), you can see that the rates between the small and tall categories are very small, as expected. But we can also note that the rates for an increase in height seems higher that the rates for a decrease in height. This suggests that the evolution tends to be directed towards greater heights. Let's test such a model. That is, let's have one rate for an increase in height, one rate for a decrease in height, and one for the unlikely change between the small and tall categories.

```
# Fit Custom2 model
(increase_fit <- fitDiscrete(seedplantstree,height,
                        model=matrix(c(0, 1, 2, 2, 0, 3, 1, 3, 0), 3)))
```

```
## GEIGER-fitted comparative model of discrete data
##  fitted Q matrix:
##                medium         small          tall
##     medium -193.61505  4.883639e+01  1.447787e+02
```

```
##     small   144.77866 -1.447787e+02  3.040558e-11
##     tall     48.83639  3.040558e-11 -4.883639e+01
##
##  model summary:
##  log-likelihood = -39.705434
##  AIC = 85.410868
##  AICc = 85.863698
##  free parameters = 3
##
## Convergence diagnostics:
##  optimization iterations = 100
##  failed iterations = 0
##  frequency of best fit = 0.18
##
##  object summary:
##  'lik' -- likelihood function
##  'bnd' -- bounds for likelihood search
##  'res' -- optimization iteration summary
##  'opt' -- maximum likelihood parameter estimates
```

You can see that the AICc for this model, 85.8636982, is lower than the ARD model (92.3069718). Therefore, this model is the best because of the economy of parameters. This is the one we will use in the ancestral character states reconstructions.

Reconstruct marginal ancestral states with Diversitree

The package `diversitree` is very useful for reconstructing ancestral states on phylogenies and we will use it later in the course. However, it is a bit complicated in the way you need to format the data.

It uses the mk-n model. M is for Markov, because the modeled process is a continuous-time Markov chain, and k because it can be generalized to any number of states (k). It is frequent to distinguish modelds for two states (mk-2) from models with more than two parameters (mk-n).

```
require(diversitree)
# Change character to numeric, because this is the format diversitree accepts
char1 <- as.numeric(height)
names(char1) <- names(height)
# Create mkn model. Note that we use equal probs for all states at the root.
lik.mkn <- make.mkn(seedplantstree, char1, k=3, control=list(root=ROOT.EQUI))
# Look at the different parameters of the model, here the transition rates
argnames(lik.mkn)
```

```
## [1] "q12" "q13" "q21" "q23" "q31" "q32"
```

```
# Now create the model using constraints. Remember that 1=medium, 2=small, and
# 3=tall. q12 is the transition rate from 1 to 2, and the symbol '~' indicates an
# equivalency. To 'q21 ~ q13' indicate that the rate q12 will be equal to q13 in
# the model.
lik.mkn.base <- constrain(lik.mkn, q21 ~ q13, q23 ~ q32, q12 ~ q31)
# Create a starting point for the search
p.mkn <- starting.point.musse(seedplantstree, 3)
# Fit the model
fit.mkn <- find.mle(lik.mkn.base, p.mkn[argnames(lik.mkn.base)])
```

```
# Model parameters
fit.mkn[1:2]
```

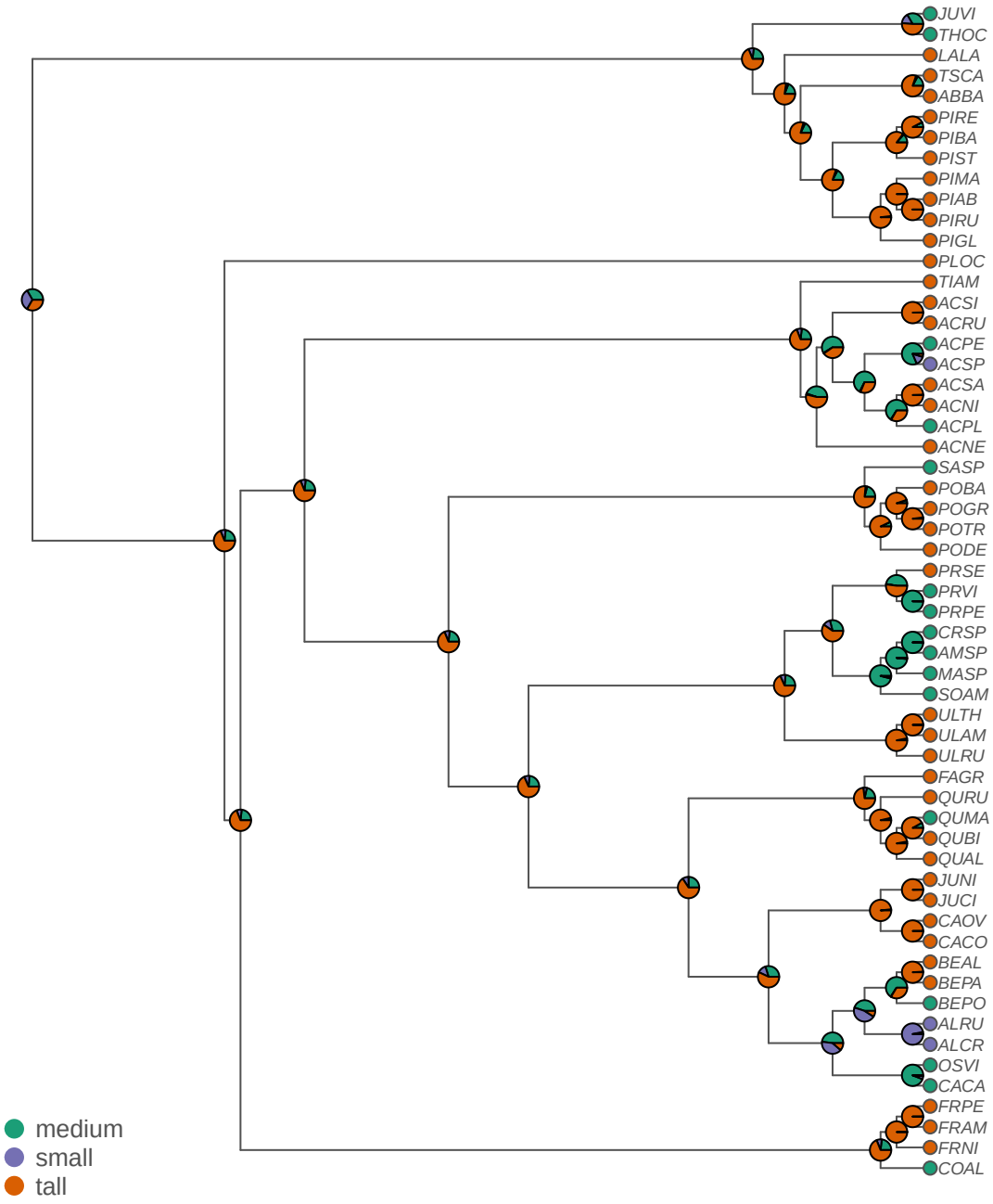
```
## $par
##          q13          q31          q32
## 1.447883e+02 4.883916e+01 1.867241e-09
##
## $lnLik
## [1] -39.70543
```

```
# Export the marginal ancestral reconstruction at the nodes of the tree
st <- t(asr.marginal(lik.mkn.base,coef(fit.mkn)))
```

We now have ancestral character estimates at each node of the tree. We could do many things with them, but we could first plot them on the phylogeny.

We can start by plotting the results with the probabilities of the different states at each node.

```
# Vector of colours
co <- c("#1b9e77","#7570b3","#d95f02")
plot(seedplantstree,type="p",FALSE,label.offset=0.6,cex=0.6,no.margin=TRUE,
     edge.color="gray32",tip.color="gray32")
tiplabels(pch=21,bg=co[as.numeric(height)],col="gray32",cex=1,adj=0.6)
nodelabels(pie=st,piecol=co,cex=0.5,col="gray32",)
legend("bottomleft",legend=levels(height),
       pch=20,col=co,bty="n",
       text.col="gray32",cex=0.8,pt.cex=2)
```



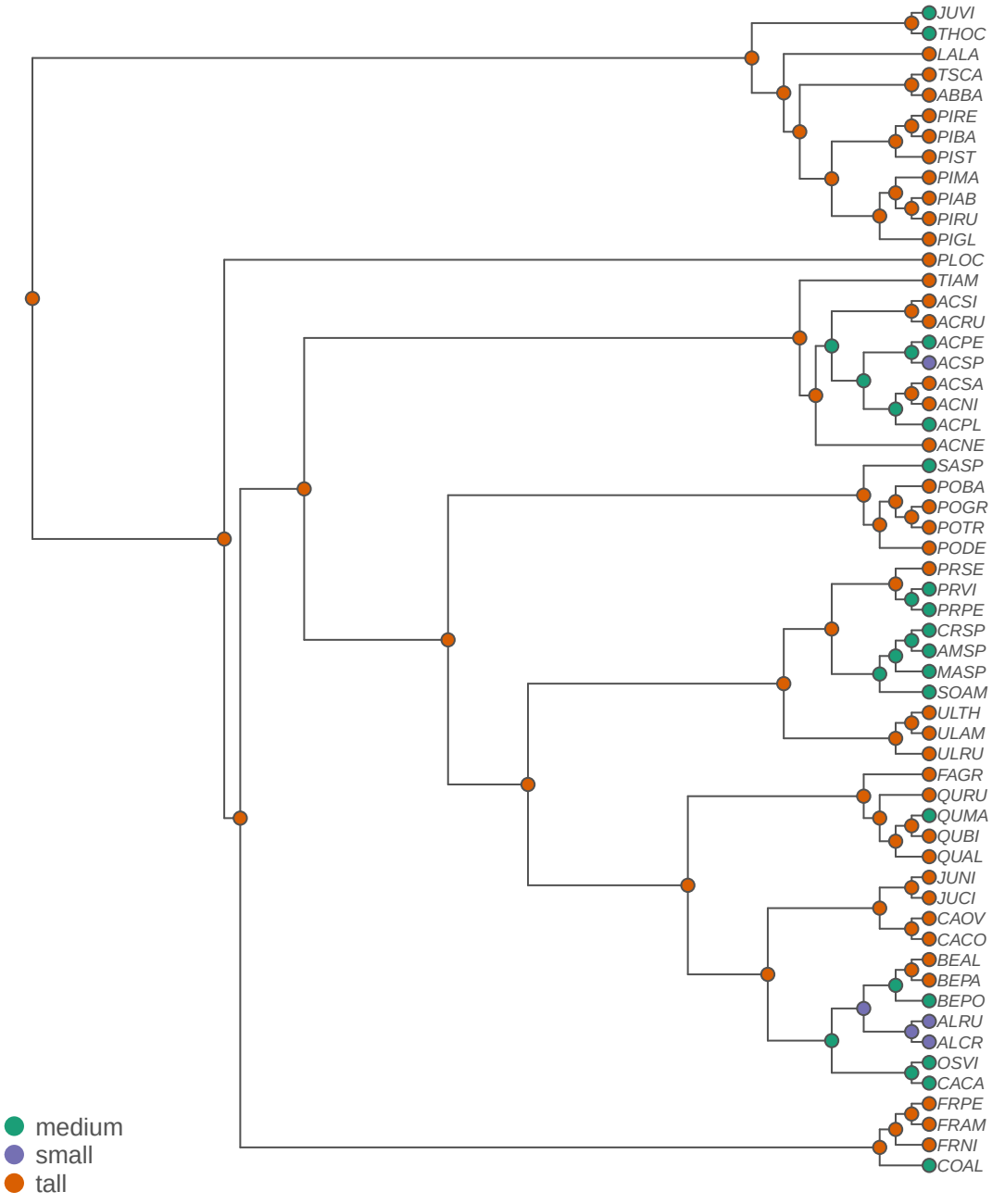
Remember that estimates at the root of the tree are unreliable. So interpret this with caution (= don't).

In some cases, it is useful to get the state with maximum probability at each node. To do this you just need to assign the state at each node with the maximum probability. This piece of code does that.

```
anc_node<-factor(character(nrow(st)),levels=levels(height))
for(i in 1:length(anc_node)) {
  anc_node[i] <- levels(height)[st[i,]==max(st[i,])]
  }
```

You can then plot the results.

```
plot(seedplantstree,type="p",FALSE,label.offset=0.6,cex=0.6,no.margin=TRUE,
     edge.color="gray32",tip.color="gray32")
tiplabels(pch=21,bg=co[as.numeric(height)],col="gray32",cex=1,adj=0.6)
nodelabels(pch=21,bg=co[as.numeric(anc_node)],cex=1,col="gray32",)
legend("bottomleft",legend=levels(height),
       pch=20,col=co,bty="n",
       text.col="gray32",cex=0.8,pt.cex=2)
```



Count the number of changes

Once you have the most likely probabilities at each node of the tree, it is possible to calculate the number of of changes between each states in the tree. As mentionned above, it is a good idea to remove the root from those estimates.

```
# Vector with tip and node states
states <- c(as.vector(height),as.vector(anc_node))
# Remove the root nodes from the comparison because ancestral states are not
# reliable
root.node <- as.numeric(names(branching.times(seedplantstree)
                [branching.times(seedplantstree)==max(branching.times(seedplantstree))]))
# Remove root node from the data
edges <- seedplantstree$edge[seedplantstree$edge[,1]!=root.node,]
# Create matrix to store the results
change.matrix <- matrix(nrow=3,ncol=3,0)
rownames(change.matrix) <- colnames(change.matrix) <- levels(height)
# Loop over all branches of the tree to count the number of changes
for (i in 1:nrow(edges)) {
  first_state <- states[edges[i,1]]
  second_state <- states[edges[i,2]]
  change.matrix[seq(1,3,1)[first_state==levels(height)],
                seq(1,3,1)[second_state==levels(height)]] <-
    change.matrix[seq(1,3,1)[first_state==levels(height)],
                  seq(1,3,1)[second_state==levels(height)]] + 1
}
# The results
change.matrix
```

```
##        medium small tall
## medium     17     2    3
## small       1     3    0
## tall        9     0   75
```

Stochastic character mapping

An alternative approach to the estimation of ancestral character state is to use a MCMC approach to sample character histories from their posterior distribution. This approach is called stochastic character mapping (Huelsenbeck et al. 2003). The idea is to simulate different instances of character evolution along the phylogeny. If you do this a large number of times, you could estimate the probabilities that a given node is at a give state.

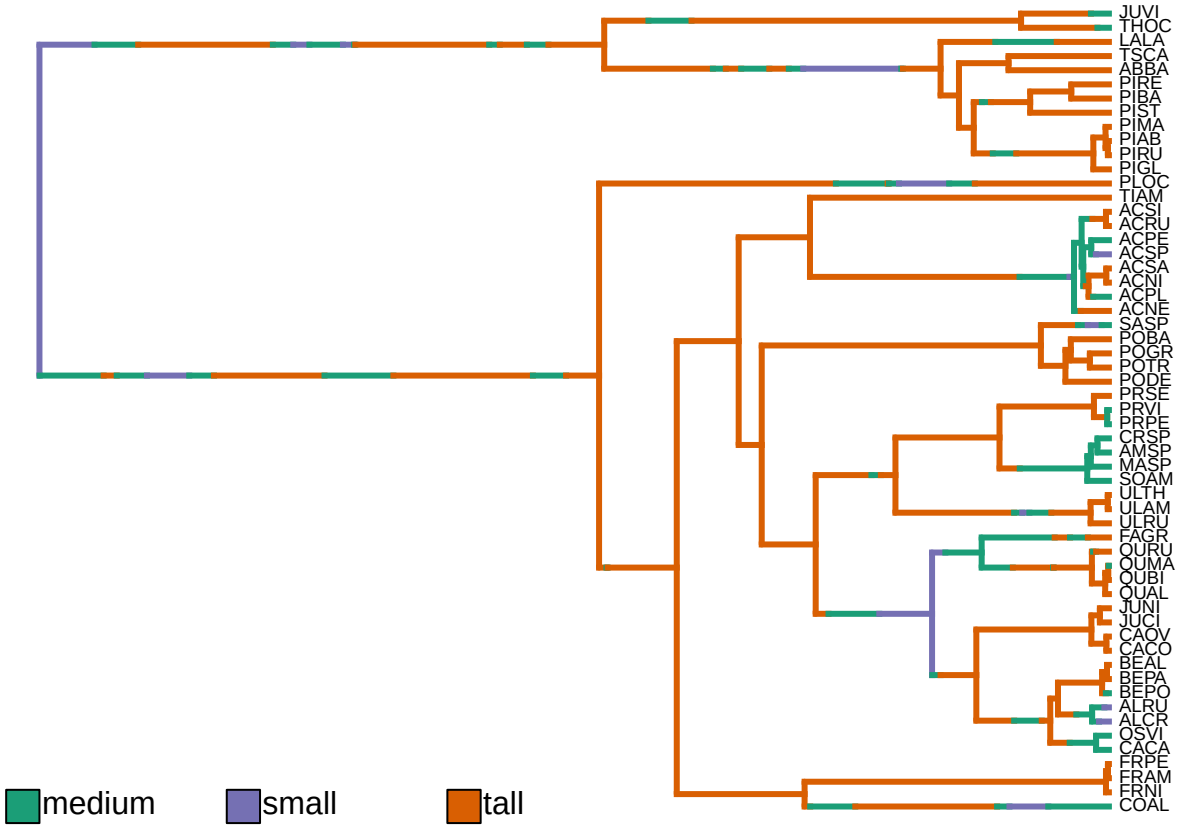
Let's generate one character map for the qualitative character height. We still use the same model as above.

```
require(phytools)
# Set seed for reproducibility
set.seed(1234)
# simulate single stochastic character map using empirical Bayes method
chartree <- make.simmap(seedplantstree, height, model=matrix(c(0, 1, 2, 2, 0, 3, 1, 3, 0), 3))
```

```
##            medium      small      tall
## medium -193.61488   48.83634 144.77854
## small   144.77854 -144.77854   0.00000
```

```
## tall     48.83634    0.00000 -48.83634
##    medium     small      tall
## 0.3333333 0.3333333 0.3333333
```

```
cols <- setNames(c("#1b9e77","#7570b3","#d95f02"), sort(unique(height)))
plotSimmap(chartree, cols, pts = FALSE, lwd = 3, fsize=0.6)
add.simmap.legend(colors = cols, vertical = FALSE, prompt = FALSE, x = 0, y = 1)
```



A single simulation is not very useful on it's own, because it is very unlikely that the character evolved exactly this way. However, if you simulate several maps, then you could use this distribution to estimate ancestral states, number of transitions, etc.

```
# Simulate 200 stochastic character maps
chartrees <- make.simmap(seedplantstree, height,
                         model=matrix(c(0, 1, 2, 2, 0, 3, 1, 3, 0), 3), nsim = 200)
```

```
## make.simmap is sampling character histories conditioned on the transition matrix
## Q =

##            medium      small      tall
## medium -193.61488   48.83634 144.77854
## small   144.77854 -144.77854   0.00000
## tall     48.83634    0.00000 -48.83634

## (estimated using likelihood);
## and (mean) root node prior probabilities
## pi =
```

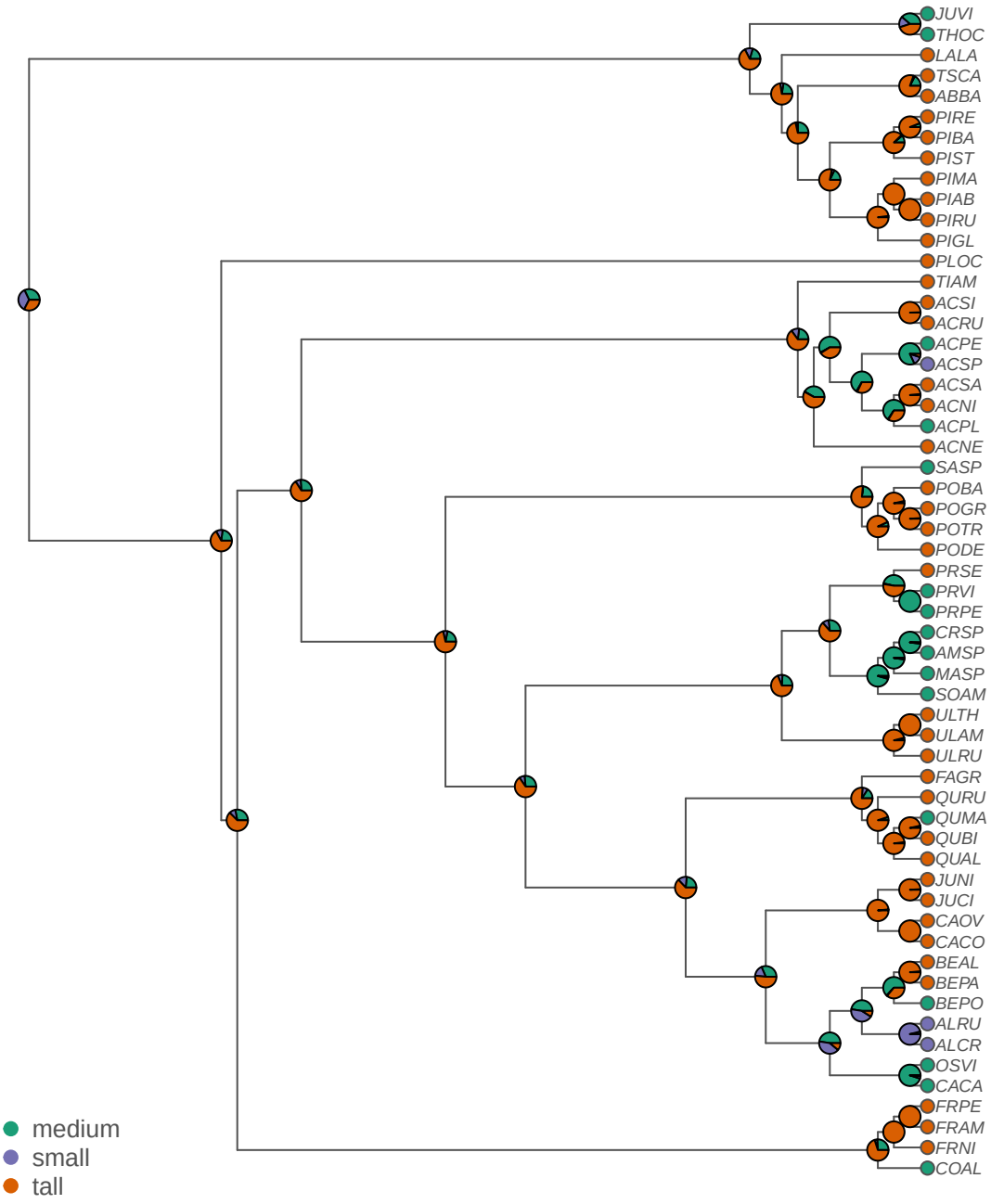
```
##    medium     small      tall
## 0.3333333 0.3333333 0.3333333
```

```
## Done.
```

```
# Output summary information
(res_simmap <- describe.simmap(chartrees, plot = FALSE))
```

```
## 200 trees with a mapped discrete character with states:
##  medium, small, tall
##
## trees have 95.255 changes between states on average
##
## changes are of the following types:
##      medium,small medium,tall small,medium small,tall tall,medium
## x->y        11.52       36.03        12.91          0      34.795
##      tall,small
## x->y          0
##
## mean total time spent in each state is:
##         medium      small      tall    total
## raw  0.2572475 0.08506466 0.6897921 1.032104
## prop 0.2492457 0.08241866 0.6683357 1.000000
```

```
# Plot the tree with posterior probabilities of states at each node
plot(seedplantstree,type="p",FALSE,label.offset=0.6,cex=0.6,no.margin=TRUE,
     edge.color="gray32",tip.color="gray32")
tiplabels(pch=21,bg=co[as.numeric(height)],col="gray32",cex=1,adj=0.6)
nodelabels(pie=res_simmap$ace,piecol=co,cex=0.5,col="gray32")
legend("bottomleft",legend=levels(height),
       pch=20,col=co,bty="n",
       text.col="gray32",cex=0.8,pt.cex=1.5)
```



These reconstructions represent the joint probability of having each state at all nodes. This is because it is based on simulations across the whole tree. You can see that the estimated states are similar to the ones obtained for marginal reconstruction, but not completely so.

Let's now count the number of changes on the tree, again removing the root.

```
# Assigne state with ML at each node
anc_node<-factor(character(nrow(res_simmap$ace)),levels=levels(height))
for(i in 1:length(anc_node)) {
  anc_node[i] <- levels(height)[res_simmap$ace[i,]==max(res_simmap$ace[i,])]
  }
states <- c(as.vector(height),as.vector(anc_node))
# Remove the root nodes from the comparison because ancestral states are not
# reliable
```

```
root.node <- as.numeric(names(branching.times(seedplantstree)
                [branching.times(seedplantstree)==max(branching.times(seedplantstree))]))
edges <- seedplantstree$edge[seedplantstree$edge[,1]!=root.node,]
# Count the number of changes across the tree
change.matrix <- matrix(nrow=3,ncol=3,0)
rownames(change.matrix) <- colnames(change.matrix) <- levels(height)
for (i in 1:nrow(edges)) {
  first_state <- states[edges[i,1]]
  second_state <- states[edges[i,2]]
  change.matrix[seq(1,3,1)[first_state==levels(height)],
                seq(1,3,1)[second_state==levels(height)]] <-
    change.matrix[seq(1,3,1)[first_state==levels(height)],
                  seq(1,3,1)[second_state==levels(height)]] + 1
}
change.matrix
```

```
##        medium small tall
## medium     19     2    3
## small       0     2    0
## tall        9     0   75
```

Parsimony reconstructions

It is possible to obtain a parsimonious character reconstruction using likelihood or stochastic mapping if we force the transition rates to be very small. This makes sense as when rates of evolution are small, there are few multiple substitutions along the branches and at the limit parsimony becomes a likelihood estimator (as for the phylogeny reconstruction). However, if there are indications in the data that some substitution rates are not very small, then likelihood is the best option.

References

Baum D., S. Smith. 2012. *Tree Thinking: An Introduction to Phylogenetic Biology*. Roberts and Company Publishers.

Gascuel O., M. Steel. 2014. Predicting the ancestral character changes in a tree is typically easier than predicting the root state. *Systematic Biology* 63: 421–435.

Huelsenbeck J.P., R. Nielsen, J.P. Bollback. 2003. Stochastic Mapping of Morphological Characters. *Systematic Biology* 52:131–158.

Kass R.E., A.E. Raftery. 1995. Bayes factor. *Journal of the American Statistical Association* 90:773–795.

Paquette A., S. Joly, C. Messier. 2015. Explaining forest productivity using tree functional traits and phylogenetic information: two sides of the same coin over evolutionary scale? *Ecology and Evolution* 5:1774–1783.

Schluter D., T. Price, A. O. Mooers and D. Ludwig. 1997. Likelihood of ancestor states in adaptive radiation. *Evolution* 51: 1699–1711.