Branch length evaluation for Phylogenetic Diversity: a worked example

Daniel R. Miranda-Esquivel

2018 - 07 - 02

Four taxa and two areas

Preparing the data space

First, we load the required libraries:

```
## cleaning
rm(list = ls())

## libraries

## installing and loading the package

##install.packages("../../blepd_0.1.1.tar.gz", repos = NULL, type="source")

library(blepd)

packageVersion("blepd")
```

```
## [1] '0.1.4.2018.7.3.2152'
## To plot trees you can use ggtree, ape or phytools. The example is based on
## ggtree as a matter of choice.

library(ggtree)

library(gridExtra)

library(RColorBrewer)
```

Now, we load the data included in the package: tree and distribution.

```
#data(package = "blepd")

## trees

data(tree)

str(tree)
```

```
## List of 5
##  $ edge       : int [1:6, 1:2] 5 6 6 5 7 7 6 1 2 7 ...
##  $ edge.length: num [1:6] 1 1 1 1 1 1
##  $ Nnode      : int 3
##  $ tip.label  : chr [1:4] "t1" "t2" "t3" "t4"
##  $ root.edge  : num 1
```

```
##  - attr(*, "class")= chr "phylo"
##  - attr(*, "order")= chr "cladewise"
initialTree <- tree


## distributions

data(distribution)

str(distribution)

##  int [1:2, 1:4] 1 0 0 1 1 1 0 0 1
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:2] "A1" "A2"
##   ..$ : chr [1:4] "t1" "t2" "t3" "t4"
dist4taxa <- distribution

## distribution to XY

distXY <- matrix2XY(dist4taxa)


## plotting

## the tree

plotTree <-  ggtree(initialTree, ladderize=TRUE,
                 color="black", size=1, linetype="dotted") +
          geom_tiplab(size=6, color="black") +
          theme_tree2() +
          labs(title = "Four terminals, equal branch length")


print(plotTree)
```
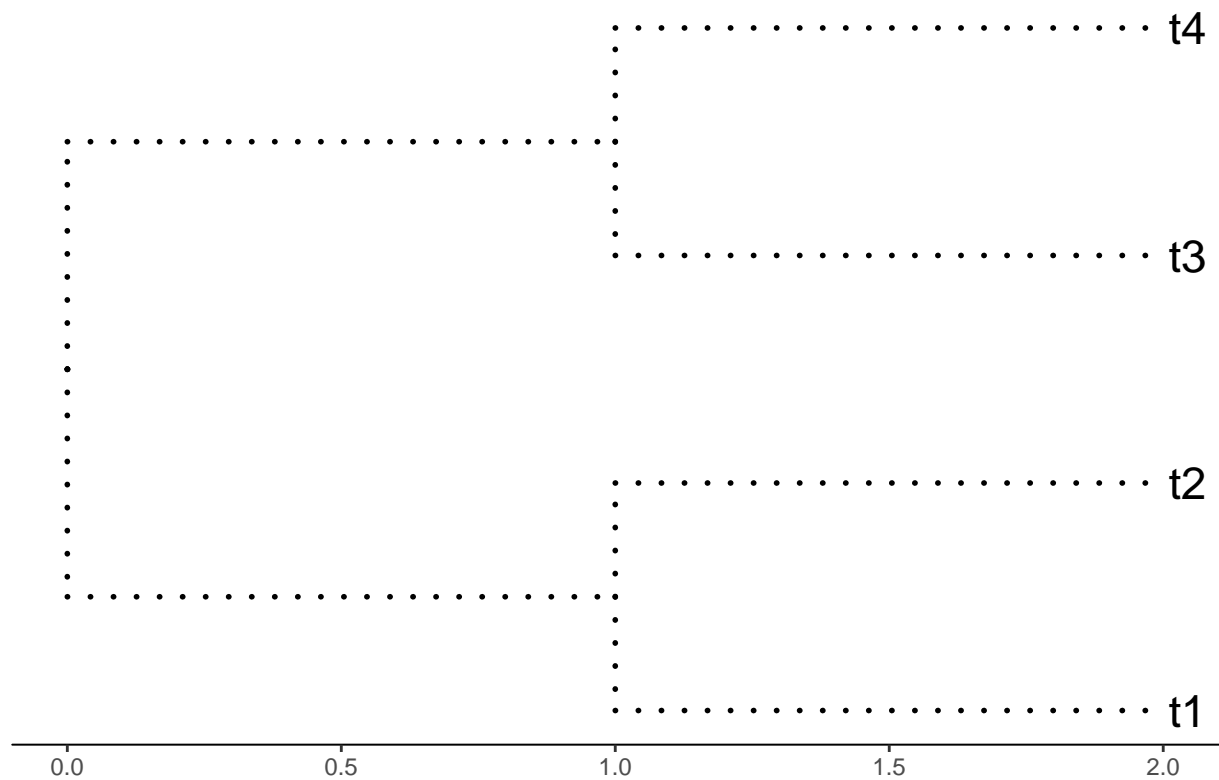
Four terminals, equal branch length


```
## the distribution

plotDistrib <- ggplot(data=distXY,
                      aes(x= Area, y= Terminal),
                      size =11) +
              geom_point() +
              labs(title = "Terminals and Distributions",
                   y = "Terminal",
                   x = "Area")

print(plotDistrib)
```
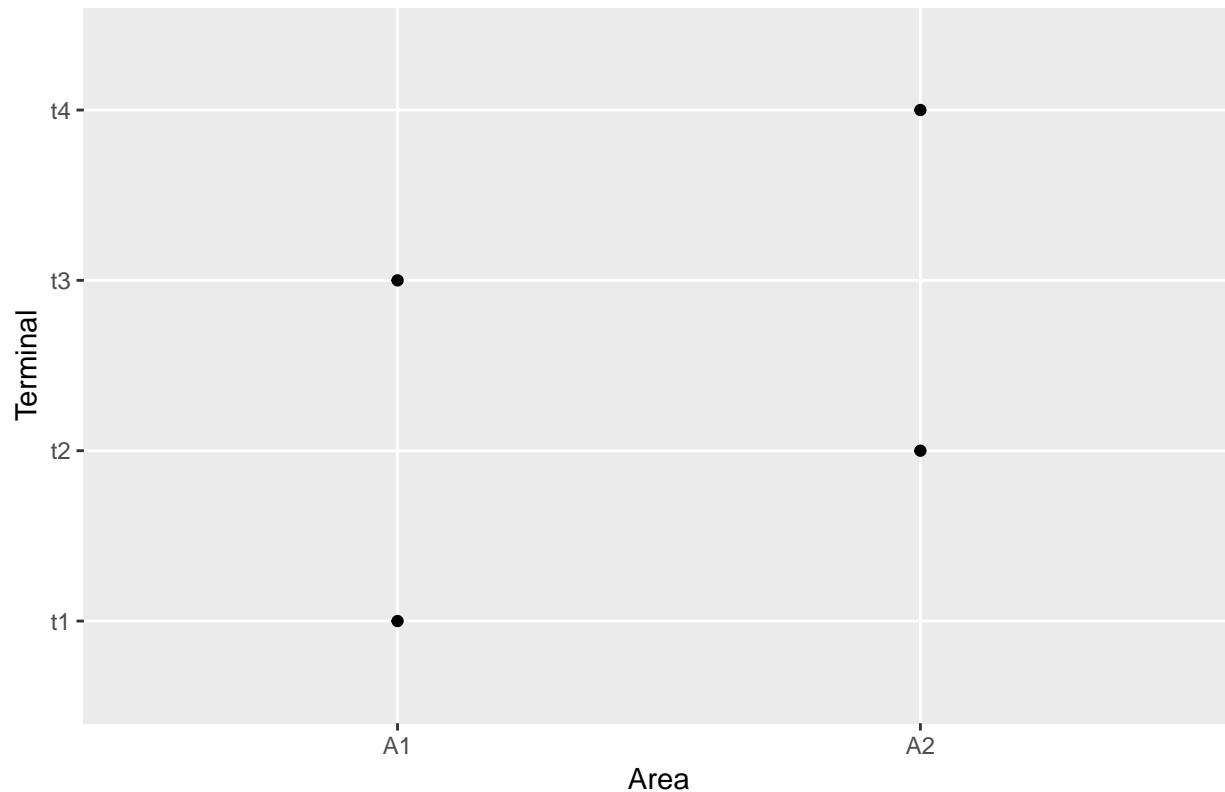
Terminals and Distributions

We check whether names in both objects: initialTree and dist4taxa are the same.

```
all(colnames(dist4taxa) == initialTree$tip.label)
```

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

We report the branch length, and calculate the PD values.

```
initialTree$edge.length
```

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

```
initialPD <- PDindex(tree=initialTree, distribution = dist4taxa)

initialPD
```

```
## [1] 4 4
```

Function to evaluate a single terminal

To test the effect of changing the branch length in a single terminal ("t1"), we will use the function *evalTerminal*. This function uses four parameters: tree, distribution, tipToEval (label of the tip), approach (two options: "lower"/"upper", to evaluate from 0 to the actual length or from the actual length to the sum of all branch lengths).

```
evalTerminal(tree = initialTree,
             distribution = dist4taxa,
             tipToEval = "t1",
             approach = "lower" )
```

```
## branchLengthChange     bestInitialArea     bestModifiedArea
##         "0.9999"             "A1A2"                 "A2"
##      initialLength
##              "1"
```

The lower limit reported when we change the branch length for terminal t1 is 0.99, therefore any change in this branch length will modify the area selected from A1A2 to A2, as the tie between the path between terminals t1/t3 (area A1) vs t2/t4 (area A2) will be solved in favour of t2/t4 when A1 is shorter.

Tree evaluation function

branch length

The function to test all terminals at the same time is *evalTree*, with two parameters: the tree and the distribution. The function returns a data.frame object with 14 fields: labelTerminal, lowerBranchLength, InitialArea, lowerFinalArea, initialLength, upperBranchLength, upperFinalArea, changeLower, changeUpper, deltaUpper, deltaLower, deltaPD, areaDelta, and abDelta.

```
finalResults <- evalTree(tree = initialTree, distribution = dist4taxa)

finalResults
```

```
##   labelTerminal InitialArea initialLength lowerFinalArea lowerBranchLength
## 1            t1        A1A2             1             A2            0.9999
## 2            t2        A1A2             1             A1            0.9999
## 3            t3        A1A2             1             A2            0.9999
## 4            t4        A1A2             1             A1            0.9999
##   changeLower deltaLower upperFinalArea upperBranchLength changeUpper
## 1          A2      1e-04             A1            1.0001          A1
## 2          A1      1e-04             A2            1.0001          A2
## 3          A2      1e-04             A1            1.0001          A1
## 4          A1      1e-04             A2            1.0001          A2
##   deltaUpper deltaPD     areaDelta abDelta
## 1      1e-04       0 L:_A2_/U:_A1       0
## 2      1e-04       0 L:_A1_/U:_A2       0
## 3      1e-04       0 L:_A2_/U:_A1       0
## 4      1e-04       0 L:_A1_/U:_A2       0
```

The extreme sensitivity of the PD results to the terminal branch length is seen in the column absolute length difference (=abDelta), as any length change -larger than 0-, will modify the area selected.

We plot the results to see the effect in each terminal, as a table.

```
plotResults <- ggplot(data=finalResults, aes(x= labelTerminal, y= initialLength,
                    shape="Actual",
                    colour=InitialArea)) +
            geom_point(size= 7) +
            geom_point(aes(x= labelTerminal, y= lowerBranchLength,
                        colour=lowerFinalArea,
                        shape="Lower_limit"), size=7) +
            labs(title = "Branch length change, lower limits. Equal branches.",
                colour = "Area selected",
                shape = "Terminal branch length value",
                y = "Terminal branch length",
                x = "Terminal")
```

```
print(plotResults)
```

Branch length change, lower limits. Equal branches.
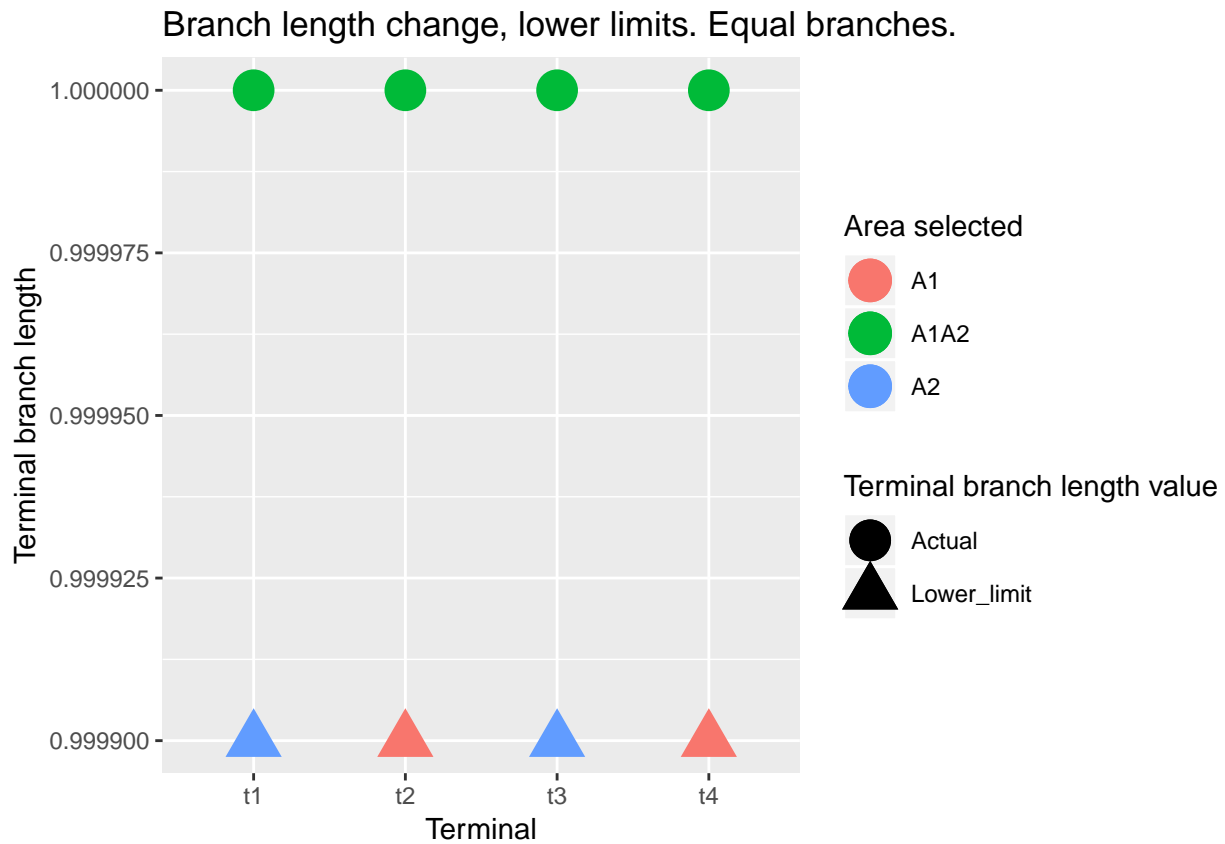
or plotted as a simple table.

```
countFreqChanges <- table(finalResults$areaDelta)


countFreqChanges <- as.data.frame(countFreqChanges, ncol=1)


colnames(countFreqChanges) <- c("Area change","Freq")


row.names(countFreqChanges) <- NULL


countFreqChanges
```

```
##    Area change Freq
## 1 L:_A1_/U:_A2    2
## 2 L:_A2_/U:_A1    2
```

or plotted into the tree:

```
theTitle <- paste("Initial area selected:",finalResults$InitialArea[1])

p0 <-    ggtree(initialTree, layout="slanted", ladderize=TRUE,
              color=c("red","blue","red","blue","black","black","black"),
```
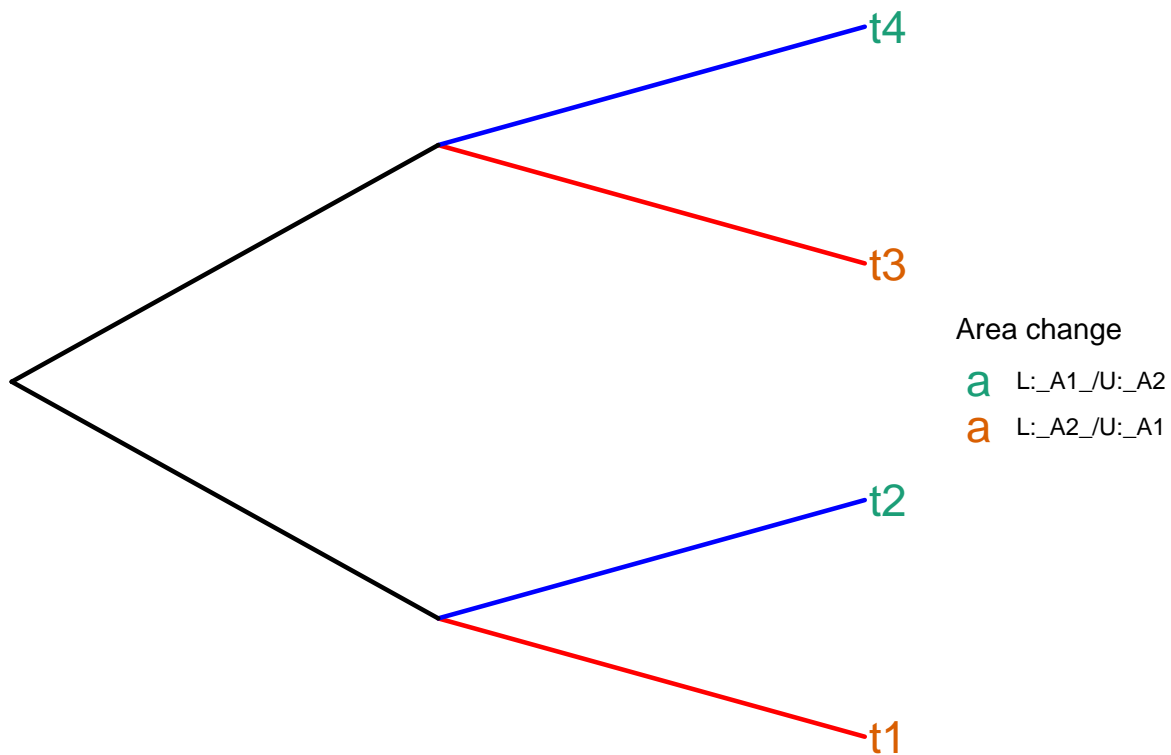
```
              size=0.8 ) +
        theme(legend.position="right") +
        labs(title = theTitle)


p <- p0 %<+% finalResults + geom_tiplab(aes(color=areaDelta), size =6) +
        scale_colour_brewer("Area change", palette="Dark2")


print(p)
```

Initial area selected: A1A2

For terminals t1/t3, a change from 1 to 0.99 in branch length -the lower limit (=L)- will change the initial area selected (A1A2) to A2; or a change from 1 to 1.01 in branch length -the upper limit(=U)-, will change the area to A1.

branch swap

A second, and different approach, is to evaluate the effect in PD when terminal branch lengths are swapped. In this case it is not the sensitivity to the branch length as a parameter, but the stability to the actual branch lengths.

The function to perform the analysis is *swapBL*, that has four parameters: the tree, the distribution, the model to evaluate (valid models are "simpleswap", "allswap" -default value- and "uniform"), and the number of times to swap (default value = 100).

Uning the default parameters we get.

```
swapBL(tree = initialTree,
       distribution = dist4taxa)
```

```
## model to test allswap reps 100
```

```
##   AreaSelected Freq
## 1       A1A2 *  100
```

As this is a tree where all branches are equal, there is no impact when the branch lengths are swapped.

Or we could use the random uniform branch length model.

```
swapBL(tree = initialTree,
       distribution = dist4taxa,
       model = "uniform")
```

```
## model to test uniform reps 100
```

```
##   AreaSelected Freq
## 1       A1A2 *  100
```

This is a tree where all branches are equal, therefore min and max are equal. There is no impact when the branch lengths are swapped, and areas A1A2 are selected.

An empirical example: *Rhynoclemmys* data.

We read the data sets: distribution and trees.

```
## read distributional data

setwd("../testData/")

distribution <- as.matrix(read.table("Rhinoclemmys_Distribution",
                                     stringsAsFactors=FALSE,
                                     header=TRUE,
                                     row.names=1,
                                     sep=",")
                          )


## trees

treeFiles <- dir(pattern=".tre")

treeFiles
```

```
## [1] "Rhinoclemmys_igrClock_exp_genus_data.nexus.con.tre"
## [2] "Rhinoclemmys_igrClock_exp_uniform_data.nexus.con.tre"
## [3] "Rhinoclemmys_NonClock_genus_data.nexus.con.tre"
```

```
RhinoclemmysData <- list()

RhinoclemmysData$distribution <- distribution

RhinoclemmysData$trees  <- lapply(treeFiles,FUN=read.nexus)

RhinoclemmysData$trees  <-lapply(RhinoclemmysData$trees,
```

```
                            FUN=root,
                            outgroup=c("Go_polyphemus","Go_agassizii"),
                            resolve.root = TRUE)
```

These three trees correspond to different clock models, and we want to test whether the clock used will have any effect in the areas chosen.

Fist we calculate the PD value for each tree, we save the values as a table and later we convert the table to a matrix:

```
RhinoclemmysData$tablePD <- lapply(RhinoclemmysData$trees,
                            FUN=PDindex,
                            distribution=RhinoclemmysData$dist,
                            root=TRUE)

RhinoclemmysData$matrixPD <- as.data.frame(
                            matrix(
                                unlist(RhinoclemmysData$tablePD),
                                nrow= length(treeFiles),
                                byrow=TRUE))
```

now, it is time to name trees, terminals and areas:

```
RhinoclemmysData$nameTrees <- rownames(RhinoclemmysData$matrixPD) <- gsub("Rhinoclemmys_","",
                                    gsub("_data.nexus.con.tre","",
                                            treeFiles))

RhinoclemmysData$nameAreas <- colnames(RhinoclemmysData$matrixPD) <-
                            rownames(RhinoclemmysData$dist)

RhinoclemmysData$nameTerminals <- colnames(RhinoclemmysData$dist)
```

now we can plot the trees

```
for(treeNumber in 1:length(treeFiles)){

cat(RhinoclemmysData$nameTrees[[treeNumber]],"\n")

    plot.phylo(RhinoclemmysData$trees[[treeNumber]],
                main=RhinoclemmysData$nameTrees[[treeNumber]])



    }
```
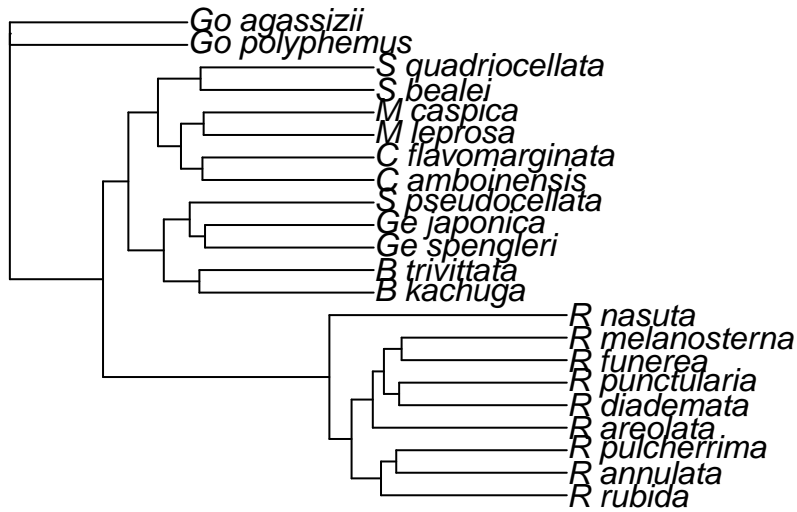
igrClock_exp_genus

igrClock_exp_genus

Go agassizii
Go polyphemus
S quadriocellata
S bealei
M caspica
M leprosa
C flavomarginata
C amboinensis
S pseudocellata
Ge japonica
Ge spengleri
B trivittata
B kachuga
R nasuta
R melanosterna
R funerea
R punctularia
R diademata
R areolata
R pulcherrima
R annulata
R rubida

igrClock_exp_uniform

igrClock_exp_uniform

Go agassizii
Go polyphemus
R nasuta
R melanosterna
R funerea
R punctularia
R diademata
R areolata
R pulcherrima
R annulata
R rubida
S quadriocellata
S bealei
M caspica
M leprosa
C flavomarginata
C amboinensis
S pseudocellata
Ge japonica
Ge spengleri
B trivittata
B kachuga
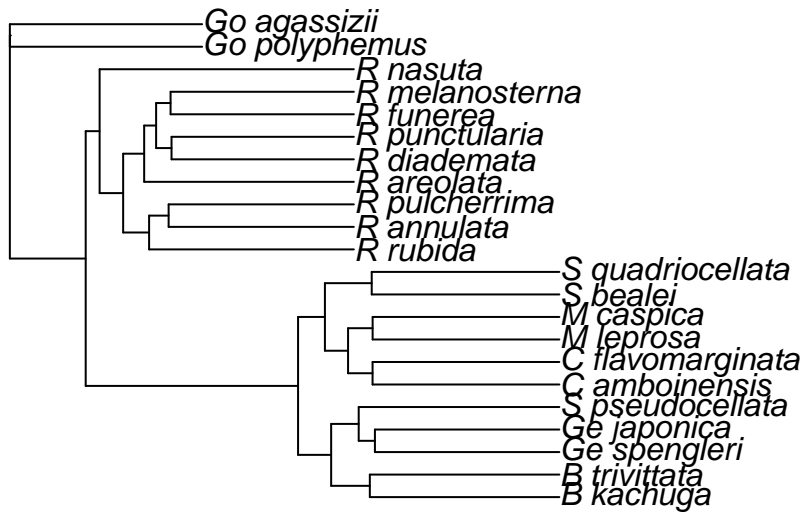
NonClock_genus

NonClock_genus

```
                    ┌─ S quadriocellata
                 ┌──┤
                 │  └─ S bealei
              ┌──┤  ┌─ M caspica
              │  └──┤
              │     └─ M leprosa
           ┌──┤     ┌─ C flavomarginata
           │  └─────┤
           │        └─ C amboinensis
        ┌──┤  ┌─ S pseudocellata
        │  └──┤  ┌─ Ge japonica
        │     └──┤
        │        └─ Ge spengleri
      ┌─┤     ┌─ B trivittata
      │ └─────┤
      │       └─ B kachuga
      │     ┌─ R nasuta
      │  ┌──┤  ┌─ R pulcherrima
      │  │  └──┤
      │  │     └─ R annulata
      │  │  ┌─ R melanosterna
      │  └──┤ ┌─ R funerea
      │     │┌┤
      │     └┤└─ R punctularia
      │      │┌─ R diademata
      │      └┤
      │       └─ R areolata
      │        └─ R rubida
    ┌─┤  ┌─ Go agassizii
    └─┤──┤
      └──── Go polyphemus
```
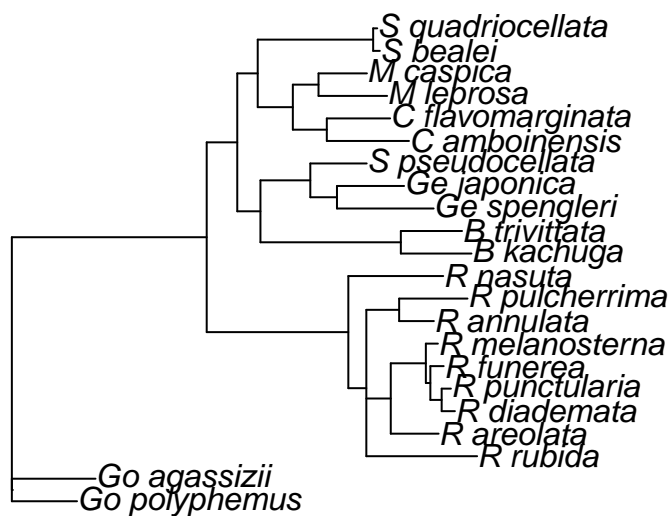
We perform some basic calculations:

```
# eval tree function


RhinoclemmysData$evalPD <- lapply(RhinoclemmysData$trees,
                          FUN=evalTree,
                          distribution=RhinoclemmysData$dist,
                          root=TRUE)



# sum of deltas
RhinoclemmysData$sumDeltas <-   sapply(RhinoclemmysData$evalPD,
                            function (x) {sum(x$abDelta/x$deltaPD)})



# max PD value


RhinoclemmysData$maxPD <- colnames(RhinoclemmysData$matrixPD)[apply(
                          RhinoclemmysData$matrixPD,1,which.max)]



# swap branch lengths, allswap


RhinoclemmysData$swapBLalls <- lapply(RhinoclemmysData$trees,
                          FUN=swapBL,
                          distribution=RhinoclemmysData$dist,
                          root=TRUE)
```

```
## model to test allswap reps 100
## model to test allswap reps 100
## model to test allswap reps 100

RhinoclemmysData$swapBLalls

## [[1]]
```

```
##   AreaSelected Freq
## 1          B *   99
## 2            F    1
##
## [[2]]
##   AreaSelected Freq
## 1          F *  100
##
## [[3]]
##   AreaSelected Freq
## 1            B    7
## 2          F *   93
```

```
RhinoclemmysData$swapBLunif <- lapply(RhinoclemmysData$trees,
                                      FUN=swapBL,
                                      distribution=RhinoclemmysData$dist,
                                      model = "uniform",
                                      root=TRUE)
```

```
## model to test uniform reps 100
## model to test uniform reps 100
## model to test uniform reps 100
```

```
RhinoclemmysData$swapBLunif
```

```
## [[1]]
##   AreaSelected Freq
## 1          B *   89
## 2            F   11
##
## [[2]]
##   AreaSelected Freq
## 1          F *  100
##
## [[3]]
##   AreaSelected Freq
## 1            B    4
## 2          F *   96
```

In this case, the tree selected changes the area selected, and we are confident in the results as in the three cases the branch swap does not affect the results.

And plotted into the trees:

```
options(warn=-1)

for(treeNumber in 1:length(treeFiles)){

cat(RhinoclemmysData$nameTrees[[treeNumber]],"\n")


theTitle <- paste("Initial area selected:",RhinoclemmysData$evalPD[[treeNumber]]$InitialArea[1])



p0 <-    ggtree(RhinoclemmysData$trees[[treeNumber]], layout="slanted", ladderize=TRUE,
              ##color=c("red","blue","red","blue","black","black","black"),
```

```
                color=c("black"),
                 size=0.5 ) +
          theme(legend.position="right") +
          labs(title = theTitle)



p <- p0 %<+% RhinoclemmysData$evalPD[[treeNumber]] +
          geom_tiplab(aes(color=areaDelta), size =5) +
          scale_colour_brewer("Area change", palette="Dark2")



print(p)


}
```
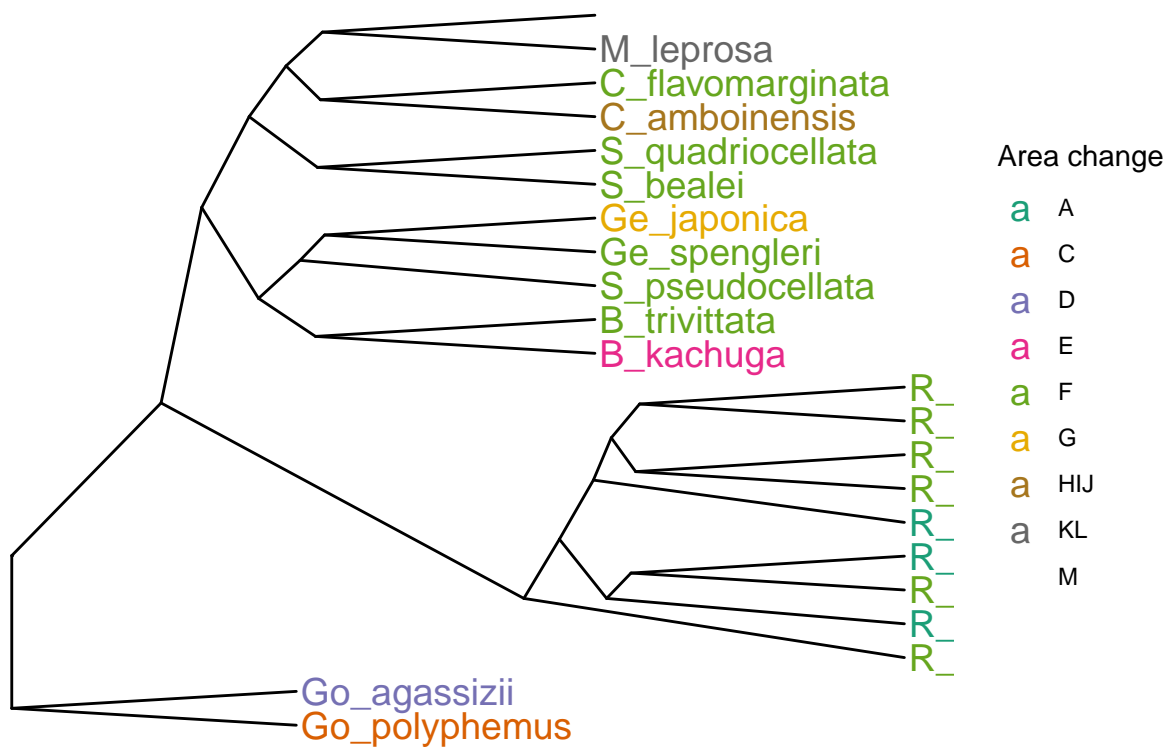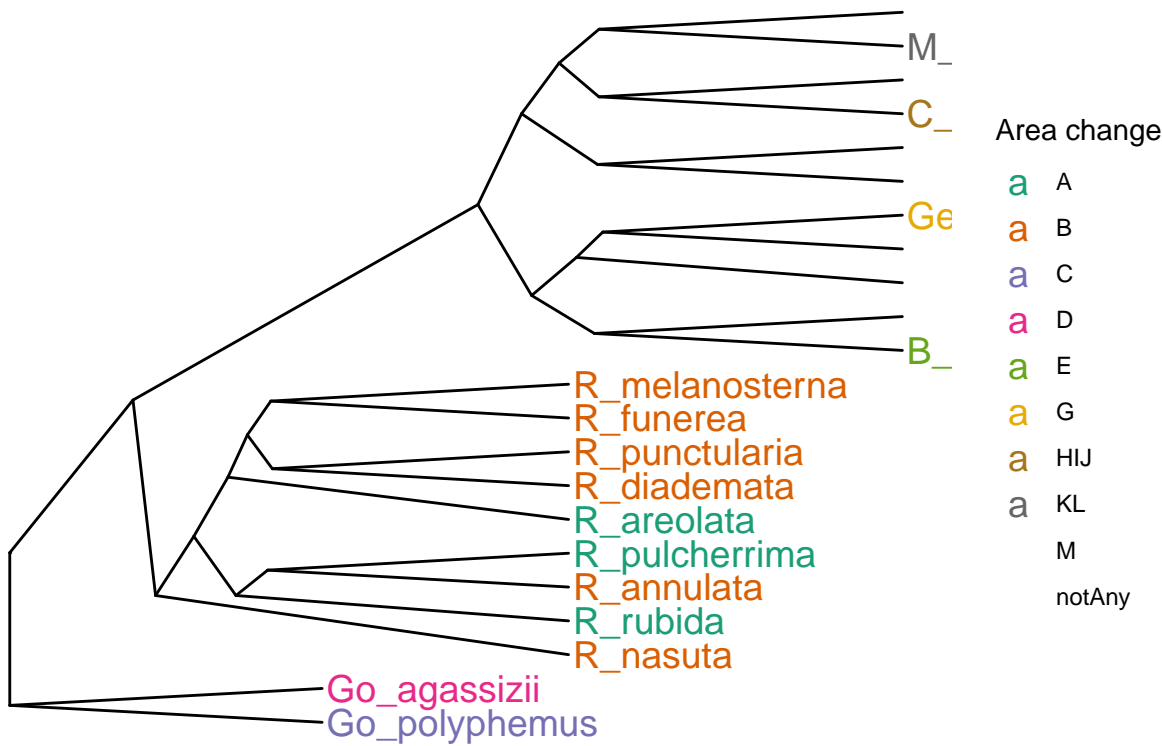
igrClock_exp_genus

Initial area selected: B

igrClock_exp_uniform

Initial area selected: F

Area change

- a A
- a B
- a C
- a D
- a E
- a G
- a HIJ
- a KL
- M
- notAny

R_melanosterna
R_funerea
R_punctularia
R_diademata
R_areolata
R_pulcherrima
R_annulata
R_rubida
R_nasuta
Go_agassizii
Go_polyphemus

NonClock_genus

Initial area selected: F

Area change

- a A
- a B
- a C
- a D
- a E
- a G
- a HIJ
- a KL
- M
- notAny

M_leprosa
C_amboine
Ge_japonic
B_k
R_pu
R_dia
R_fune
R_mela
R_areo
R_p
R_annu
R_
R_nas
Go_agassizii
Go_polyphemus