Branch length evaluation for Phylogenetic Diversity: the algorithm

Daniel R. Miranda-Esquivel

2018 - 04 - 09

The Algorithms

Constructing the functions

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)
```

```
## Loading required package: ape
## Loading required package: picante
## Loading required package: vegan
## Loading required package: permute
##
## Attaching package: 'permute'
## The following object is masked from 'package:devtools':
##
##     check
## Loading required package: lattice
## This is vegan 2.5-2
## Loading required package: nlme
## Loading required package: methods
```

```
packageVersion("blepd")
```

```
## [1] '0.1.4.2018.7.5.1625'
```

```
## to plot trees

library(ggtree)
```

```
## Loading required package: ggplot2
## Loading required package: treeio
```

```
##
## Attaching package: 'treeio'

## The following objects are masked from 'package:ape':
##
##     drop.tip, Nnode, Ntip

## ggtree v1.10.2  For help: https://guangchuangyu.github.io/ggtree
##
## If you use ggtree in published research, please cite:
## Guangchuang Yu, David Smith, Huachen Zhu, Yi Guan, Tommy Tsan-Yuk Lam. ggtree: an R package for visua

##
## Attaching package: 'ggtree'

## The following object is masked from 'package:nlme':
##
##     collapse

## The following object is masked from 'package:ape':
##
##     rotate
```

```
library(gridExtra)

library(RColorBrewer)
```

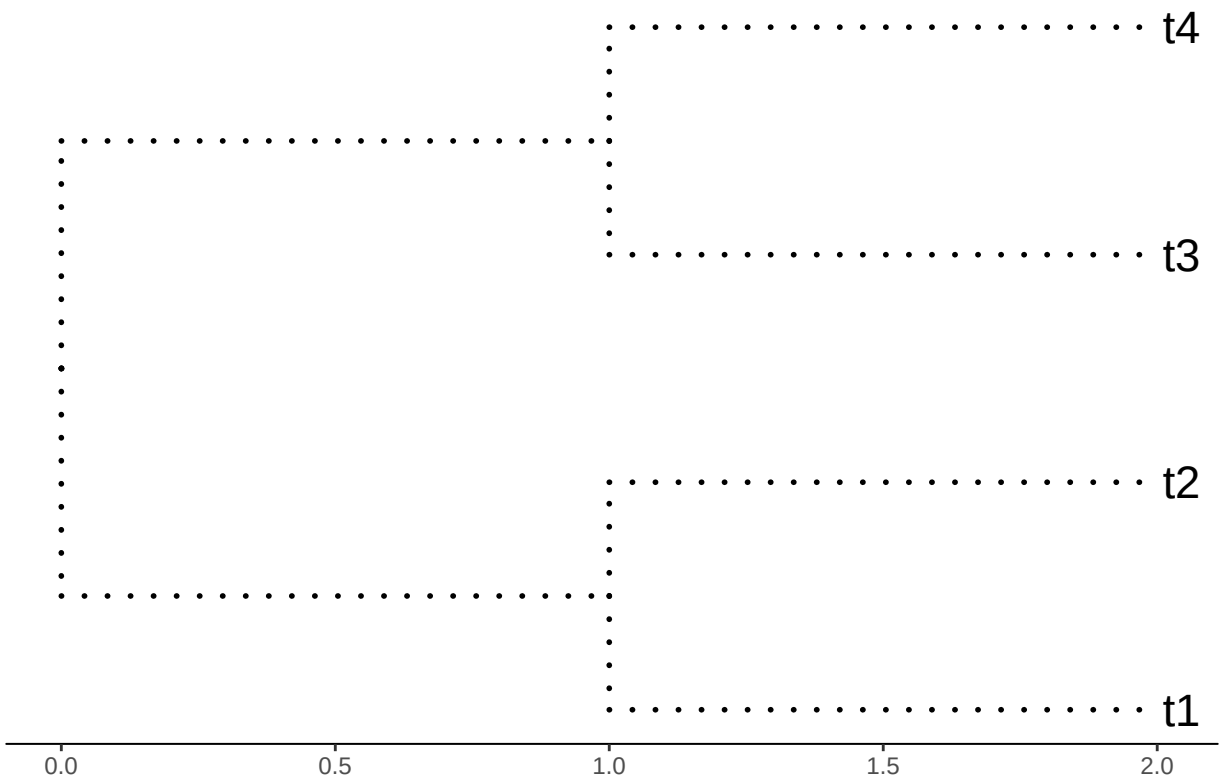
Now, we load the data: tree and distributions

```
## A binary tree with four terminals, newick format.

initialTree <- read.tree("../testData/tree")

str(initialTree)
```

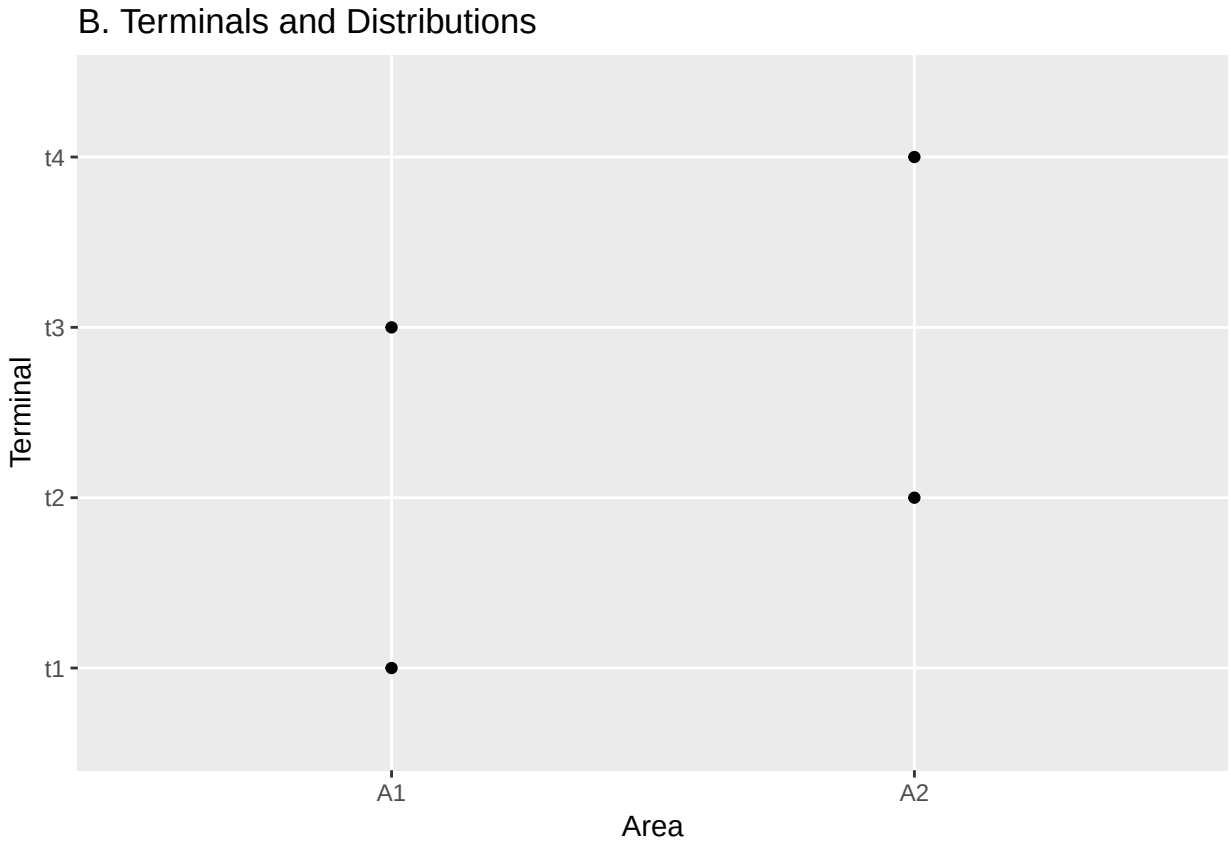
```
## List of 5
##  $ edge       : int [1:6, 1:2] 5 6 6 5 7 7 6 1 2 7 ...
##  $ edge.length: num [1:6] 1 1 1 1 1 1
##  $ Nnode      : int 3
##  $ tip.label  : chr [1:4] "t1" "t2" "t3" "t4"
##  $ root.edge  : num 1
##  - attr(*, "class")= chr "phylo"
##  - attr(*, "order")= chr "cladewise"
```

```
## distributions for four taxa and two area, table format.

dist4taxa <- as.matrix(read.table("../testData/distribution",
                                  stringsAsFactors=FALSE,
                                  header=TRUE,
                                  row.names=1)
                       )

## distribution to XY

distXY <- matrix2XY(dist4taxa)

### plotting:

## A. the tree
```

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

print(plotTree)
```




```
## B. the distribution

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

plotDistrib
```



We check whether names in both objects, trees and distributions are the same:

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

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

We report all branch lengths 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
```

Single taxon evaluation function

To test the effect of changing a single terminal branch length, we will:

1. create a copy of the initial branch length.
2. calculate the initial PD and get the area(s) with the max value.
3. change the length of a given terminal: t1.
4. recalculate PD and get the area(s) with the max value.
5. compare both results (from steps 3 to 4) to evaluate the effect of the perturbation.

```r
bestInitialArea <- row.names(dist4taxa)[which(initialPD == max(initialPD))]

bestInitialArea
```

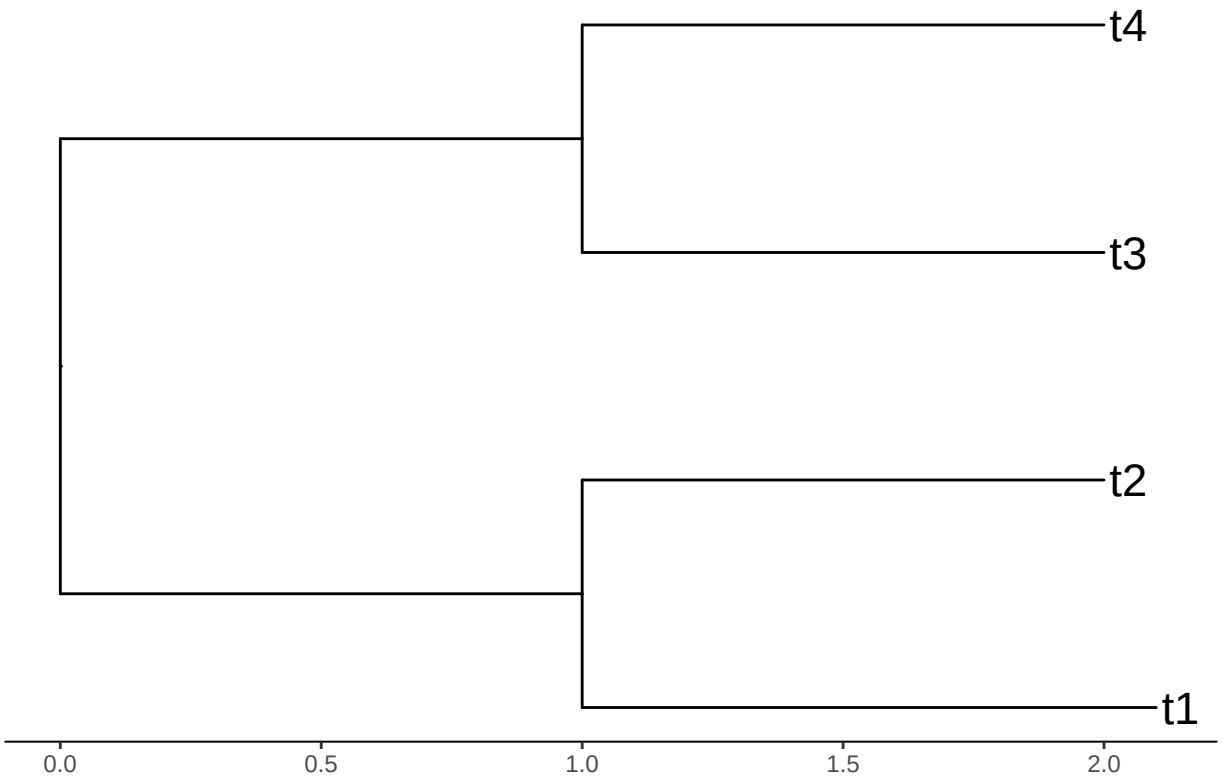
```
## [1] "A1" "A2"
```

```r
tipToEval <- "t1"

value <- 1.1

numberTipToEval <- which(initialTree$tip.label %in% tipToEval)

newTree <- initialTree

## a table, binding tree$edge and tree$edge.length

createTable <- function(tree = tree){allDataTable <- tree$edge
                                     allDataTable <- cbind (allDataTable, tree$edge.length)
                                    return(allDataTable)
                  }


## new tree, with t1 branch changed to "value"

newTree$edge.length[which(createTable(initialTree)[,2] %in% numberTipToEval)] <- value

newTree$edge.length
```

```
## [1] 1.0 1.1 1.0 1.0 1.0 1.0
```

```r
## plotting

plotNewTree <- ggtree(newTree) +
               theme_tree2() +
               geom_tiplab(size=6, color="black") +
               labs(title = "C. Four terminals, non equal branch length")

print(plotNewTree)
```

C. Four terminals, non equal branch length




```
### PD for the modified tree

modifiedPD <- PDindex(tree = newTree, distribution = dist4taxa)

bestModifiedArea <- row.names(dist4taxa)[which(modifiedPD == max(modifiedPD))]

## comparing initial and modified areas selected

bestInitialArea
```

```
## [1] "A1" "A2"
```

```
bestModifiedArea
```

```
## [1] "A1"
```

```
bestInitialArea == bestModifiedArea
```

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

As shown, a modification in the branch length will impact the area selected, we can consider that the results are *very* sensible to changes in branch length.

THE function

To evaluate the behaviour os a single terminal branch length we must:

1. modify a branch length,
2. recalculate the PD value and
3. compare the effect of this perturbation.

We can turn these steps into a function called *evalTerminal*, with four parameters:

1. a tree with branch lengths
2. the distribution object
3. the terminal to evaluate
4. the approach to evaluate the terminal, that could be "lower", or "upper" limits, to evaluate the minimal or the maximal value of the branch length where the PD value changes, therefore another area is selected.

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

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

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

```
evalTerminal(tree = initialTree,
             distribution = dist4taxa,
             tipToEval = "t2",
             approach = "lower" )
```

```
## branchLengthChange    bestInitialArea   bestModifiedArea
##           "0.9999"              "A1A2"               "A1"
##      initialLength
##                "1"
```

A similar result will arrive from changing the terminal branch t2, but in this case the tie is solved to favour A1.

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

```
## branchLengthChange    bestInitialArea   bestModifiedArea
##           "1.0001"              "A1A2"               "A1"
##      initialLength
##                "1"
```

And, we will get the same result by changing the branch length of the terminal t1 from 1 to 1.x, to find the upper limit. The tie is solved in favour of area A2, opposite to the solution when we found the lower limit. In this case, even the smaller change in any terminal branch value, will modify the results.

We test the effect of the branch length for all terminals.

```
newTree
```

```
##
## Phylogenetic tree with 4 tips and 3 internal nodes.
##
## Tip labels:
## [1] "t1" "t2" "t3" "t4"
##
```

```
## Rooted; includes branch lengths.
```

```
modifiedPD
```

```
## [1] 4.1 4.0
```

```
bestModifiedArea
```

```
## [1] "A1"
```

```
evalTerminal(tree = newTree, distribution = dist4taxa, tipToEval = "t3", approach = "upper" )
```

```
##                maxPD      bestInitialArea       unModifiedArea
## "0.0999999999999996"                 "A1"                  "*"
##        initialLength
##                  "1"
```

```
evalTerminal(tree = newTree, distribution = dist4taxa, tipToEval = "t3", approach = "lower" )
```

```
## branchLengthChange    bestInitialArea   bestModifiedArea
##           "0.8999"                "A1"               "A2"
##      initialLength
##                "1"
```

The THING is the PD difference between areas, and whether this value could be accumulated in a single terminal branch, or if the contribution to the PD value is evenly distributed among all terminal branches, and therefore to change the PD value more than one terminal branch length must have to change, to select another area.

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.

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

```
##   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 change the area selected.

Null model tests

We could also swap branch lengths and evaluate the behaviour before and after swapping. This swapping could be done by swapping two terminal branch lengths *simpleswap*, all terminal branch lengths *allswap* or replacing the terminal branch length with randomly distributed values, using an uniform distribution -*uniform*-, where thwe min and max values are obtained from the actual branch lengths.