Simulated tree - Feb 2016

S. Le Vu
(Dated: February 18, 2016)

```
library(ape)
library(ggplot2)
```

I. LOAD STUFF

Import simulation outputs

```
## Load newest Rdata
l <- list.files(pattern="*.Rdata") # list.files(pattern="Rdata$") list.files(pattern="out")
load(l[length(l)])
ls()

[1] "l"     "o"     "parms" "tree"
```

Import ExaML tree 0

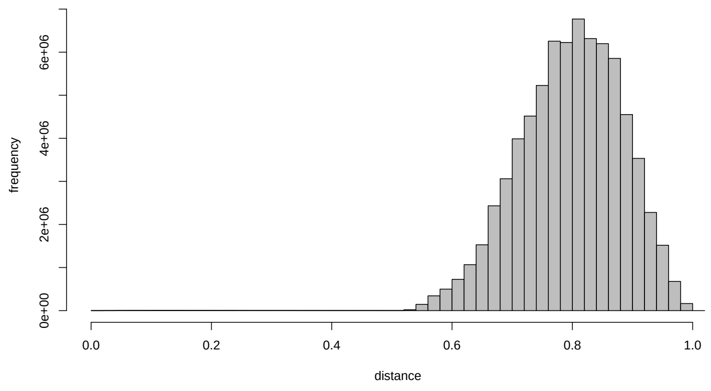
```
t <- read.tree(file = "../phylo-uk/data/ExaML_result.subUKogC_noDRM.finaltree.000")
## drop OG
og <- c("Ref1", "Ref2", "Ref3", "Ref4", "Ref5", "Ref6", "HXB2")
t <- drop.tip(t, og )
t


Phylogenetic tree with 12164 tips and 12163 internal nodes.

Tip labels:
34695, 21677, 72292, 81292, 85197, 53538, ...

Rooted; includes branch lengths.
```

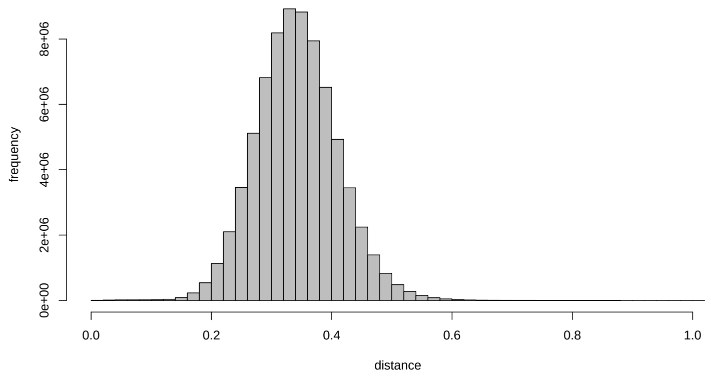
II. TREE DISTANCES

Patristic distances ?

```
####  cluster size to real data. Need to have same number of clusters...

## get distances
##- matrix first into distances
tree


Phylogenetic tree with 12164 tips and 10671 internal nodes.

Tip labels:
7, 8, 9, 10, 11, 12, ...

Unrooted; includes branch lengths.
```

```
simtree <- as.dist(cophenetic.phylo(tree))
uktree <- as.dist(cophenetic.phylo(t))
head(simtree)

[1] 24930 24930 24930 24930 24930 24930

head(uktree)

[1] 2.000001e-06 8.505254e-02 7.145780e-02 1.257209e-01 1.010726e-01 1.104545e-01

## normalize
simx <- simtree / (max(simtree) - min(simtree))
ukx <- uktree / (max(uktree) - min(uktree))
# rm(simtree, uktree)

##- histogram distances
# summary(x)
hist(simx, breaks = 50, xlab = "distance", ylab = "frequency", main = "", col = "grey")
hist(ukx, breaks = 50, xlab = "distance", ylab = "frequency", main = "", col = "grey")
```





III. CLUSTERING

Clustering with fixed number of groups ?

```
simhc <- hclust(simx, method = "average") # UPGMA
ukhc <- hclust(ukx, method = "average")
```

```r
##- cut based on height
if (F){
#- function of heights
nheights <- 10 # number of threshold
up <- round(mean(simx), 1) # cut up to the mean
# breaks <-  seq(1/nbreaks, 1-(1/nbreaks), by = 1/nbreaks)
simclus <- cutree(simhc, h = seq(up / nheights, up, by = up / nheights) ) # h = breaks
up <- round(mean(ukx),1) # cut up to the mean
ukclus <- cutree(ukhc,  h = seq(up / nheights, up, by = up /nheights) )
# rm(simx, ukx)
}

#- cut as function of k groups
kgroups <- c(500, 1000, 5000, 8000, 10000)
simclus <- cutree(simhc, k = kgroups ) # h = breaks
ukclus <- cutree(ukhc,  k = kgroups )
# colnames(simclus) <- paste("k",colnames(simclus),sep='')
# colnames(ukclus) <- paste("k",colnames(ukclus),sep='')
head(simclus)
```

```
   500 1000 5000 8000 10000
7    1    1    1    1     1
8    2    2    2    2     2
9    3    3    3    3     3
10   4    4    4    4     4
11   5    5    5    5     5
12   6    6    6    6     6
```

```r
head(ukclus)
```

```
      500 1000 5000 8000 10000
34695   1    1    1    1     1
21677   1    1    1    1     1
72292   1    1    2    2     2
81292   1    1    1    3     3
85197   1    1    3    4     4
53538   1    1    4    5     5
```

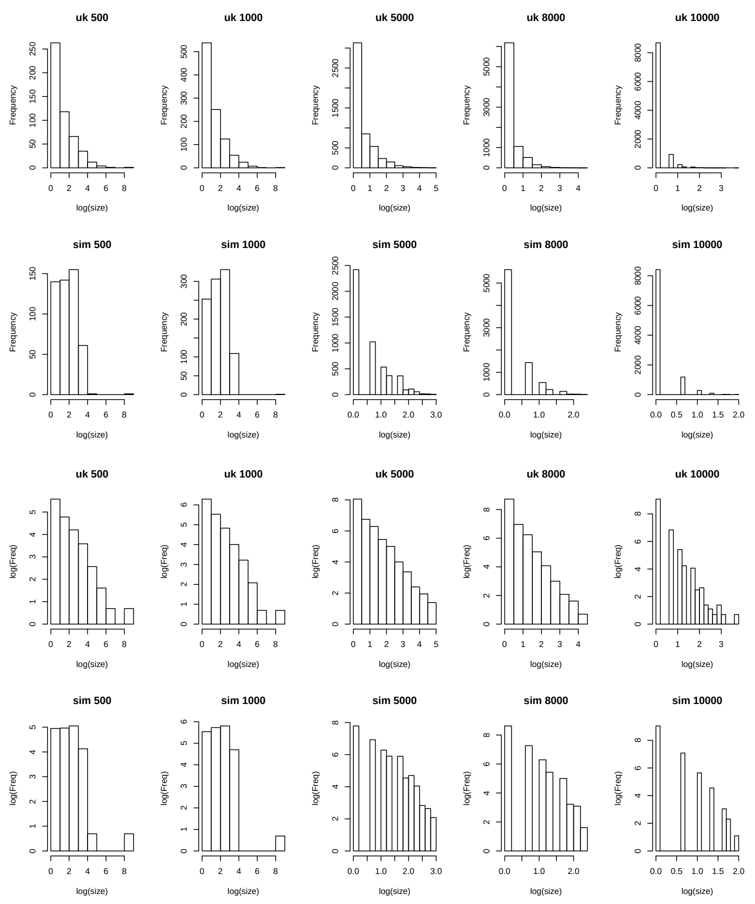
```r
##- Calculate size(=Freq) of each cluster across different threshold
simfreqClust <- apply(simclus, 2, function(x) as.data.frame(table(x))) # list
ukfreqClust <- apply(ukclus, 2, function(x) as.data.frame(table(x)))
# str(simfreqClust)
# head(simfreqClust[[1]])

##- number of different clusters by threshold # if number varies !
# sapply(simfreqClust, function(x) dim(x)[1])
# sapply(ukfreqClust, function(x) dim(x)[1])

##- cluster size
#- sim
sapply(simfreqClust, function(x) summary(x$Freq))
```

```
            500    1000   5000  8000 10000
Min.       1.00    1.00  1.000  1.00 1.000
1st Qu.    2.00    2.00  1.000  1.00 1.000
Median     6.00    6.00  2.000  1.00 1.000
Mean      24.33   12.16  2.433  1.52 1.216
3rd Qu.   13.00   13.00  3.000  2.00 1.000
Max.    7395.00 3019.00 20.000 11.00 7.000
```

```
#- uk
sapply(ukfreqClust, function(x) summary(x$Freq))

            500    1000    5000  8000  10000
Min.       1.00    1.00   1.000  1.00  1.000
1st Qu.    1.00    1.00   1.000  1.00  1.000
Median     2.00    2.00   1.000  1.00  1.000
Mean      24.33   12.16   2.433  1.52  1.216
3rd Qu.    7.00    6.00   2.000  1.00  1.000
Max.    6868.00 3486.00 142.000 86.00 39.000

##- percentiles
# sapply(freqClust, function(x) round(quantile(x$Freq,
# probs = c(0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 0.95, 0.99, 1))))
#
```

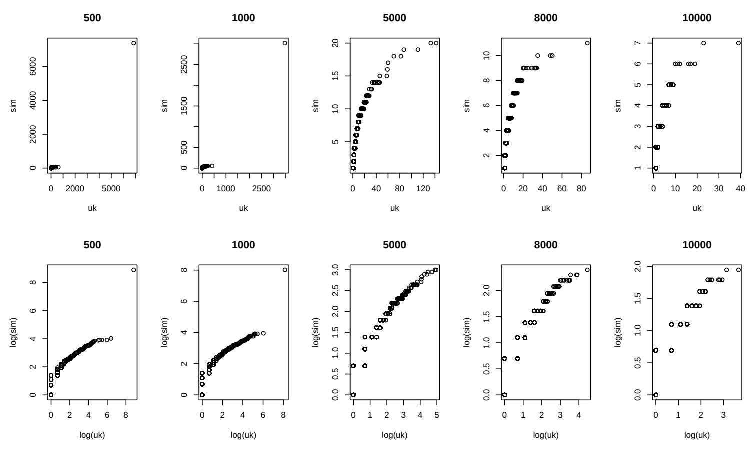
Distribution of sizes - untransformed, transformed

```
##- distr of cluster sizes: log(x) and Y untransformed
par(mfcol=c(2, length(kgroups)))
for (i in 1:length(kgroups)){
  h <- hist(log(ukfreqClust[[i]]$Freq),
       main = paste("uk", names(ukfreqClust)[i]),
       xlab = "log(size)")
  hist(log(simfreqClust[[i]]$Freq),
       main = paste("sim", names(simfreqClust)[i]),
       xlab = "log(size)")
}

##- distr of cluster sizes: log(x) and log(y)
par(mfcol=c(2, length(kgroups)))
for (i in 1:length(kgroups)){
  h <- hist(log(ukfreqClust[[i]]$Freq), plot = F)
  h$counts <- log1p(h$counts) # log(y)
  plot(h, ylab = "log(Freq)",
       main = paste("uk", names(ukfreqClust)[i]),
       xlab = "log(size)")

  h <- hist(log(simfreqClust[[i]]$Freq), plot = F)
  h$counts <- log1p(h$counts) # log(y)
  plot(h, ylab = "log(Freq)",
          main = paste("sim", names(simfreqClust)[i]),
          xlab = "log(size)")

}
```



QQ plot

```
par(mfcol=c(2, length(kgroups)))
for (i in 1:length(kgroups)){
  qqplot(ukfreqClust[[i]]$Freq,
         simfreqClust[[i]]$Freq,
         main = names(ukfreqClust)[i],
         xlab = "uk", ylab = "sim")

  qqplot(log(ukfreqClust[[i]]$Freq),
         log(simfreqClust[[i]]$Freq),
         main = names(ukfreqClust)[i],
         xlab = "log(uk)", ylab = "log(sim)")

}
```



IV. ADD PATIENTS DATA

Add data from sample states and time

```
##- converting sample states in table of co-variates ?
demes <- as.vector(read.csv(file = "demes.csv")$x)
sampleTimes <- scan( file = 'sampleTimes' )
ss  <- matrix( scan( file = 'sampleStates' ) ,
               byrow = TRUE,
               ncol = length(demes))
colnames(ss) <- demes
dim(ss)

[1] 12164   121

max(ss[,121]) # nothing on source

[1] 0

demo <- data.frame()
for (i in 1:dim(ss)[1]){ # dim(ss)[1]
  deme <- names(which(ss[i,] == 1)) # name of column which has value 1
  patient <- i
  time <- sampleTimes[i]
  age <- as.numeric( regmatches( deme,
                regexec( "\\.age([0-9])", deme) )[[1]][2] )
  care <- as.numeric( regmatches( deme,
                regexec( "care([0-9])", deme) )[[1]][2] )
  stage <- as.numeric( regmatches( deme,
                regexec( "stage([0-9])", deme) )[[1]][2] )
  risk <- as.numeric( regmatches( deme,
                regexec( "riskLevel([0-9])", deme) )[[1]][2] )
  demo <- rbind(demo, cbind(
    patient, time, age, care, stage, risk))
}
str(demo)

'data.frame': 12164 obs. of  6 variables:
 $ patient: num  1 2 3 4 5 6 7 8 9 10 ...
 $ time   : num  11245 8627 6679 8446 8900 ...
 $ age    : num  3 3 4 4 4 3 4 3 3 4 ...
 $ care   : num  1 1 1 1 1 1 1 1 1 1 ...
 $ stage   : num  1 1 1 5 4 2 5 2 3 5 ...
 $ risk    : num  1 1 1 1 1 1 1 1 2 1 ...

##- date of diagnosis ?
date0 <- as.Date('1979-01-01')
demo$datediag <- date0 + demo$time
min(demo$datediag)

[1] "1996-11-15"

max(demo$datediag)

[1] "2013-02-15"
```

Add cluster size by patient

```
####... and outdegrees
```