The coalescent model practice in R

author: Marcelo Gehara

install packages

install.packages(“phyclust”)

install.packages(“ape”)

install.packages(“phytools”)

load libraries

```
library(phyclust)
```

```
## Loading required package: ape
```

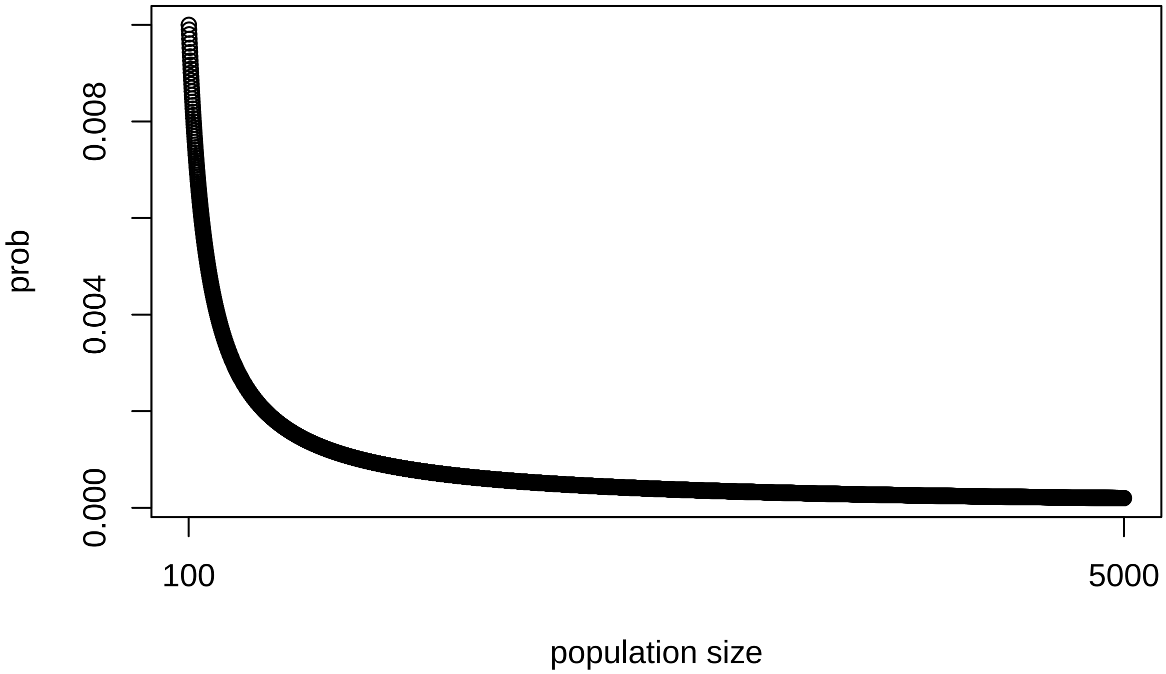
```
library(ape)
library(phytools)
```

```
## Loading required package: maps
```

Probability of coalescence of two alleles in pops with different sizes.

I’m creating a function with two arguments: (i) the Ne as a numeric vector and (ii) the ploidy. The function will iterate over all Ne values in the vector. It calculates the probability of 2 alleles coalescing in a population for an Ne value using the equation *1/Ne x ploidy*.

```
Ne.coal.prob <- function(Ne, ploidy) {
   Ne <- 1/(Ne*ploidy)
   return(Ne)
   }

## generating a vector of populations sizes
pop.sizes <- c(100:5000)

## running the function
prob <- Ne.coal.prob(Ne = pop.sizes, ploidy = 1)

## plotting the function
plot(prob, xlab="population size", xaxt = "n")
     axis(1, at=c(0,length(prob)), labels=c(min(pop.sizes),max(pop.sizes)))
```



Simulation of coalescent trees

We can simulate coalescent trees in R using the ms simulator (Hudson 2001), which is included as a function of the *phyclust r-package*. We have to assume a value of population size and a mutation rate. *ms* works in the coalescent scale so we have to convert this assumed values to Theta which is a parametric measure of population size. For haploid organisms the conversion is: *theta = 2 X Ne X mutation rate*. For diplois is: *theta = 4 X Ne X mutation rate*.

Let's simulate trees under different population sizes and compare the time to the most recent common ancestor, that is the total time for all the alleles to coalesce.

```
Ne1 = 1000 ### population size
Ne2 = 10000 ### population size
µ = 0.00001 ### mutation rate
theta1 = 2*Ne1*µ ### theta = 2Neµ (transformation to coalescent scale)
theta2 = 2*Ne2*µ
```

The *ms* function needs 3 arguments. Run *?ms* to see arguments description.

```
### Simulation
sims1 <- ms(nsam=10, nreps=1000, opts=paste('-L -T -t',theta1))
sims2 <- ms(nsam=10, nreps=1000, opts=paste('-L -T -t',theta2))

###### visualize the ns output
sims1[1:3]
```

```
## [1] "ms 10 1000 -L -T -t 0.02 "
## [2] "//"
## [3] "(((s1: 0.127218455076,(s7: 0.030603347346,s9: 0.030603347346): 0.096615105867): 0.014697760344,
```

Now we need to read the simulated trees and transform the length of the tree to generations. We read the trees using the *read.tree* function of *ape r-package*. The lengths of the branches are measured in proportion of population size. Since theta = 2Ne according to our previous conversion, we can multiply these proportions times 2Ne to get the time in generations. We use a *for* loop to do the same conversion over all trees.

```
     #### read the coalescent trees
     trees1 <- read.tree(text = sims1)
     trees2 <- read.tree(text = sims2)

     ### see edge lengths
     trees1[[1]]$edge.length
```

```
##  [1] 0.548411608 0.014697760 0.127218455 0.096615106 0.030603347
##  [6] 0.030603347 0.005317599 0.087587222 0.049011391 0.038140394
## [11] 0.010870996 0.010870996 0.002017394 0.134581223 0.134581223
## [16] 0.544895589 0.145432249 0.145432249
```

```
     for(i in 1:length(trees1))
       {
       trees1[[i]]$edge.length <- trees1[[i]]$edge.length*2*Ne1
       }

     for(i in 1:length(trees2))
     {
       trees2[[i]]$edge.length <- trees2[[i]]$edge.length*2*Ne2
     }

     ### see edge lengths
     trees1[[1]]$edge.length
```

```
##  [1] 1096.823215   29.395521  254.436910  193.230212   61.206695
##  [6]   61.206695   10.635197  175.174445   98.022781   76.280788
## [11]   21.741992   21.741992    4.034787  269.162446  269.162446
## [16] 1089.791179  290.864497  290.864497
```

Now we need to get the node heights instead of the branch lengths. We use the *nodeHeights* function to get that. Since we are interested in the time to the most recent common ancestor (tmrca) we need to get the node with the maximum height. We use the *max* function to get that.

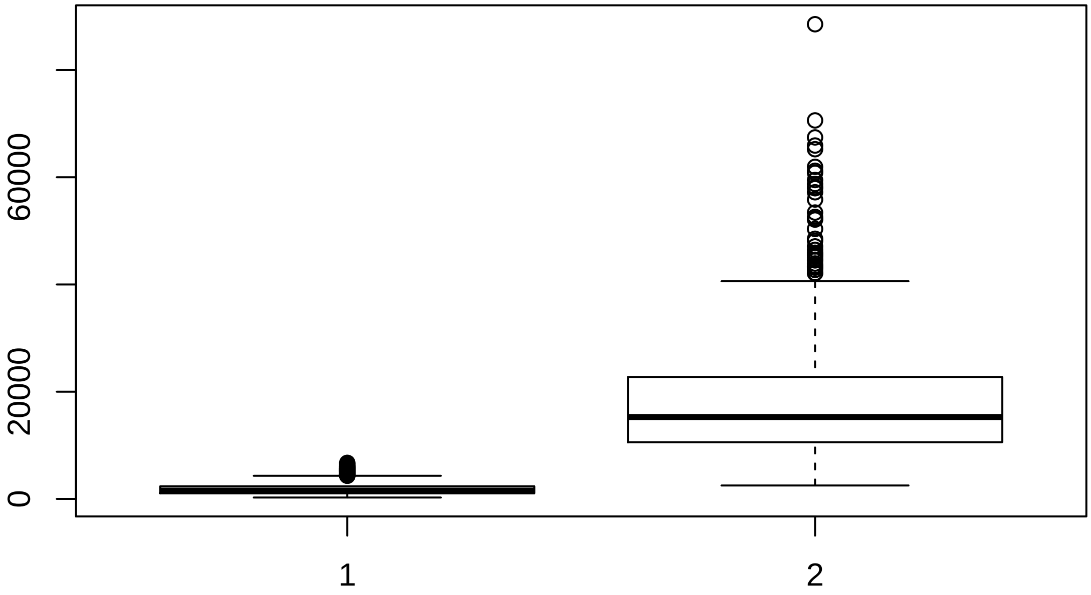
```
### tmrca of all genealogies
tmrca1 <- NULL
  for(i in 1: length(trees1)){
    tmrca1 <- c(tmrca1, max(nodeHeights(trees1[[i]])))
  }

tmrca2 <- NULL
  for(i in 1: length(trees2)){
    tmrca2 <- c(tmrca2, max(nodeHeights(trees2[[i]])))
  }

     ### mean time to the most recent common ancestor on average
     c(mean(tmrca1), sd(tmrca1))
```

```
## [1] 1824.379 1086.804
```

```
c(mean(tmrca2), sd(tmrca2))
```

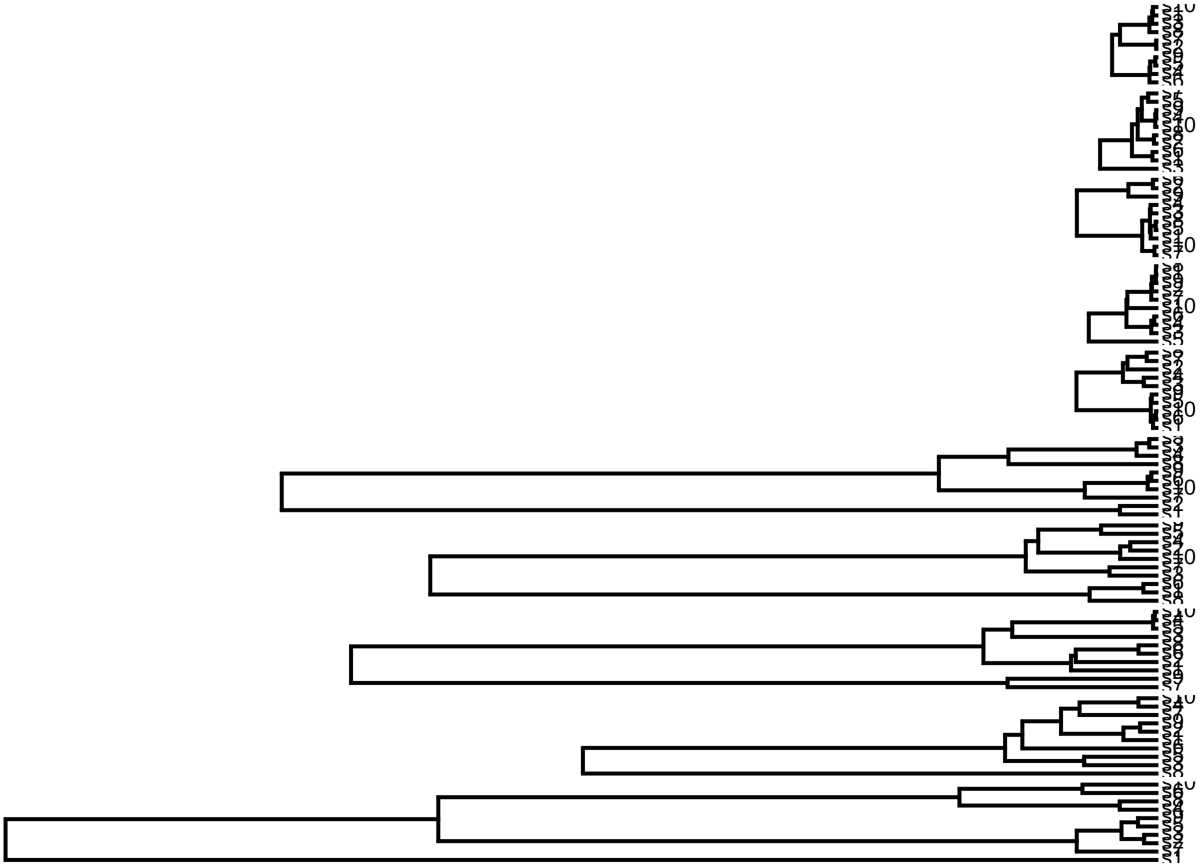
```
## [1] 18137.54 10998.37
```

```
boxplot(tmrca1,tmrca2) #### boxplot of distributions of tmrca
```



Lets plot 5 trees of each simulation side by side so we can compare the heights. The first five are from simulation 1.

```
#### plot 10 genetrees, 5 from each simulation
par(mfrow = c(10, 1))

x<-trees1[sample(1:length(trees1),5)]
x<-c(x,trees2[sample(1:length(trees2),5)])

h<-sapply(x, function(x) max(nodeHeights(x)))
l<-h-max(h)
for(i in 1:10){
     plotTree(x[[i]], xlim=c(l[i], h[i]))
}
```



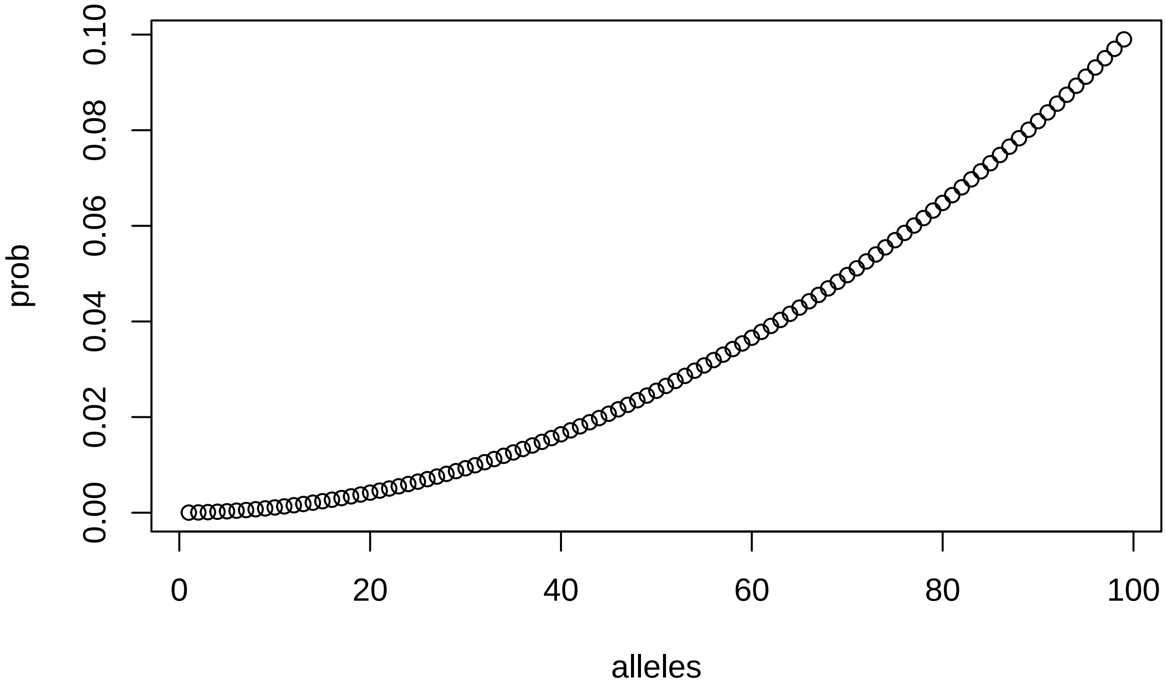
Influence of the number of alleles in the probability of coalescence.

As we've seen, the probability of coalescence increases with the number of alleles. Bellow is a function to calculate probabilities for different number of alleles. It takes three arguments. (i) The population size, (ii) the number of alleles and (iii) the ploidy. It calculates the probabilities from the total number of alleles down to the last coalescence event.

```r
al.coal.prob <- function(Ne, alleles, ploidy)
{
  alleles = 2:alleles
  pr = (alleles*(alleles-1))/(2*Ne*ploidy)
  return(pr)
}

# run the function
coal.pr <- al.coal.prob(Ne = 50000, alleles = 100, ploidy = 1)

# sort the probabilities
coal.pr <- sort(coal.pr, decreasing = F)
min(coal.pr)
```

```
## [1] 2e-05
```

```r
# plot result
plot(coal.pr, ylab="prob", xlab="alleles")
```



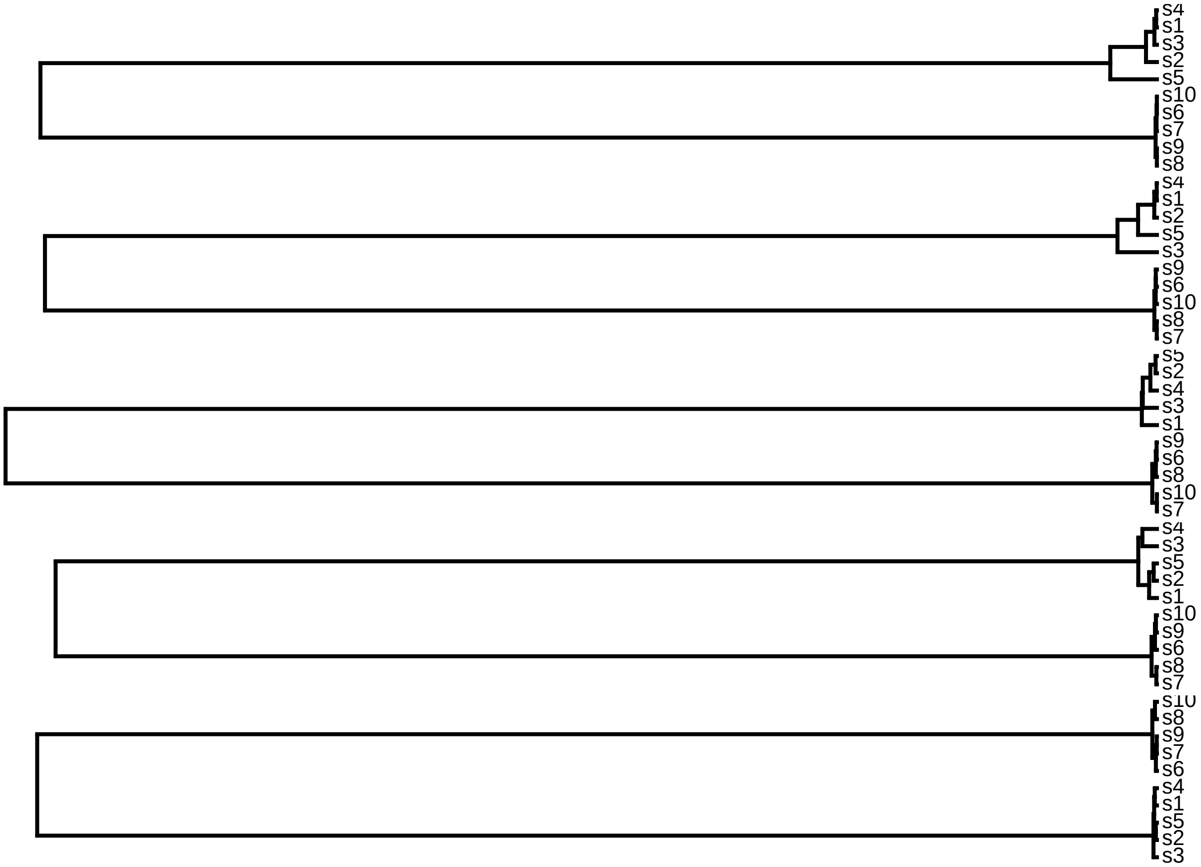
Simulations of 2 diverging diploid populations. This is similar to what we did above, but now we are simulating diverging populations. We will see the influence of migration and divergence time in the sorting of alleles.

This code

```
Ne = 10000 ### population size
µ = 0.00001 ### mutation rate
theta = 4*Ne*µ ### theta
divergence.time = 1000000 ### generations
time = divergence.time/(4*Ne) ### calescent scaled divergence
mig = 0 # 4Nm

######### simulate data
sims = ms(nsam = 10, nreps = 1000, opts=paste('-T -t ',theta,' -I 2 5 5 ',mig,
                                              ' -n 2 0.1 -ej ',time,' 2 1', sep=""))
### read simulated trees
trees <- read.tree(text = sims)
names(trees) <- NULL

#### check if the tree is sorted for pop 1
x <- unlist(lapply(trees, is.monophyletic, paste("s",1:5, sep="")))

### check proportion of sorted trees
length(which(x==T))/1000
```

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

```
### plot trees
par(mfrow = c(5, 1))
for(i in 1:length(trees))
  {
    trees[[i]]$edge.length <- trees[[i]]$edge.length*4*Ne
  }

x <- trees[sample(1:length(trees),10)]

h <- sapply(x, function(x) max(nodeHeights(x)))

l<-h-max(h)
for(i in 1:5){
  plotTree(x[[i]], xlim=c(l[i], h[i]))
}
```



End