Modelling Evolutionary Trees
CSC8622

Pedro Pinto da Silva Alexander Kell

November 5, 2016

Part 1

Question (i)

```
buildTree = function(n=10, lambda=0.5) {
  nspecies = (2*n - 2)
  cols = c("parent", "child", "birth", "termination", "length")

  tree = matrix(NA, nrow = nspecies, ncol = length(cols))
  colnames(tree) = cols
  tree[,c("parent", "child")] = 0

  t = 0
  for(k in 1:(n-1)) {
    if(k == 1) {
      parent = 2*n - 1
    } else {
      candidates = which( ! (tree[, "child"] %in% tree[, "parent"]))
      # Length of candidates is always > 1 otherwise we would
      #  have to be careful with the behavior of sample
      #  (undesired behavior for length == 1)
      parent = sample(candidates, 1)
      t = t + rexp(1, rate = k*lambda)
    }

    childs = sample(which(tree[,"parent"]==0), 2)
    tree[childs, "child"] = childs
    tree[childs, "parent"] = parent
    tree[childs, "birth"] = t
    if(k > 1) {
      tree[parent, "termination"] = t
    }
  }
  t = t + rexp(1, rate = n*lambda)
  tree[is.na(tree[, "termination"]), "termination"] = t

  tree[, "length"] = tree[, "termination"] - tree[, "birth"]
  return(tree)
}
```

```r
isExtant = function(tree, index=1:nrow(tree)) {
  ! (tree[index, "child"] %in% tree[, "parent"])
}

loadSpecies = function(path="../aux/species.txt") {
  species = read.table(path, header=FALSE, sep = "+", stringsAsFactors = FALSE)$V1
  species[-which(species=="unavailable")]
}

# Yet Another Yule (YAY)
yay = function(n=10, lambda=0.5) {
  tree = buildTree(n)
  species = loadSpecies()
  if(length(species) > 0) {
    nomes = species[sample(1:length(species), nrow(tree)+1)]
  } else {
    nomes = paste("poney", 1:(nrow(tree)+1), sep="")
  }

  yule = data.frame(Parent      = tree[, "parent"],
                    ParentName  = nomes[tree[, "parent"]],
                    Child       = tree[, "child"],
                    ChildName   = nomes[tree[, "child"]],
                    isExtant    = isExtant(tree),
                    Birth       = tree[, "birth"],
                    Termination = tree[, "termination"],
                    Length      = tree[, "length"])
  yule[yule$Parent == 2*n-1, ]$ParentName = nomes[2*n-1]
  return(yule)
}

yule = yay()
head(yule, 1)
```

```
##   Parent     ParentName Child                    ChildName isExtant
## 1      3 Neotis denhami     1 Acrantophis madagascariensis     TRUE
##      Birth Termination    Length
## 1 2.001884    2.715523 0.7136386
```

```r
yule[, -c(2,4)]
```

```
##    Parent Child isExtant    Birth Termination     Length
## 1       3     1     TRUE 2.001884    2.715523 0.71363860
## 2      13     2    FALSE 1.870345    2.052698 0.18235305
## 3      13     3    FALSE 1.870345    2.001884 0.13153915
## 4      18     4     TRUE 1.601258    2.715523 1.11426487
## 5       6     5     TRUE 0.956221    2.715523 1.75930171
## 6      19     6    FALSE 0.000000    0.956221 0.95622103
## 7      14     7     TRUE 2.513802    2.715523 0.20172074
## 8      19     8    FALSE 0.000000    1.504358 1.50435802
## 9       3     9     TRUE 2.001884    2.715523 0.71363860
## 10     12    10     TRUE 2.676007    2.715523 0.03951549
## 11     12    11     TRUE 2.676007    2.715523 0.03951549
## 12      8    12    FALSE 1.504358    2.676007 1.17164923
## 13      8    13    FALSE 1.504358    1.870345 0.36598697
## 14      2    14    FALSE 2.052698    2.513802 0.46110396
```

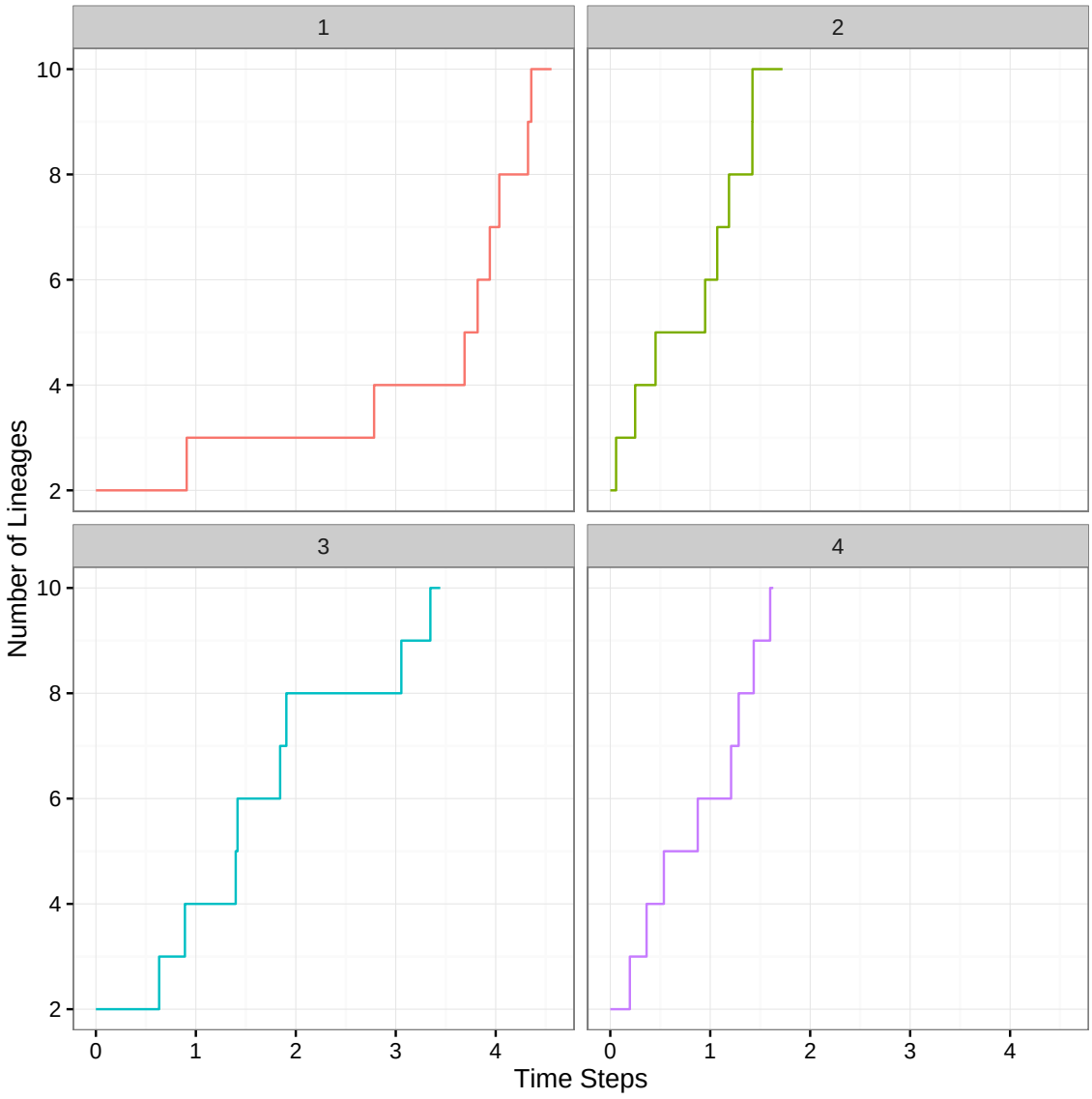
```
## 15     18    15     TRUE 1.601258    2.715523 1.11426487
## 16      2    16     TRUE 2.052698    2.715523 0.66282470
## 17     14    17     TRUE 2.513802    2.715523 0.20172074
## 18      6    18    FALSE 0.956221    1.601258 0.64503684
```

Question (ii)

```
yuleSteps = function(yule) {
  tstep = unique(sort(yule$Birth))
  tstep = c(tstep, max(yule$Termination))
  return(data.frame(tstep=tstep, nlineages=c(2:length(tstep), length(tstep))))
}

yuleSteps(yule)

##       tstep nlineages
## 1  0.000000         2
## 2  0.956221         3
## 3  1.504358         4
## 4  1.601258         5
## 5  1.870345         6
## 6  2.001884         7
## 7  2.052698         8
## 8  2.513802         9
## 9  2.676007        10
## 10 2.715523        10
```

Question (iii)

```
n = 10
lambda=0.5
four_yays = lapply(1:4, function(i) yuleSteps(yay(n, lambda)))
four_yays = rbind.fill(four_yays)
four_yays$group = ((as.numeric(rownames(four_yays)) - 1) %/% n) + 1

ggplot(four_yays, aes(x = tstep, y = nlineages))   +
  geom_step(aes(colour=factor(group))) +
  facet_wrap(~ group, ncol=2) +
  ylab("Number of Lineages") +
  xlab("Time Steps") +
  theme_bw() +
  scale_colour_discrete(guide = FALSE)
```



Part 2

Question (i)

At this stage, we introduced a new function to relabel the edges as required and return a phylo object instead.

```
# Yet Another Phylo
yaPhylo = function(n=10, lambda=0.5) {
  yule = yay(n, lambda)

  # Relabelling the nodes
  yule[yule$isExtant==TRUE, ]$Child = 1:n
  yule[yule$isExtant==FALSE, ]$Child = (n+2):(2*n-1)
```

```
  yule$Parent = yule$Child[yule$Parent]
  yule[is.na(yule$Parent), ]$Parent = n + 1

  phylo = list(edge = matrix(c(yule$Parent, yule$Child), ncol = 2),
               edge.length = yule$Length,
               tip.label = paste("t", 1:10, sep=""),
               Nnode = n - 1)
  class(phylo) = "phylo"
  return(phylo)
}
```

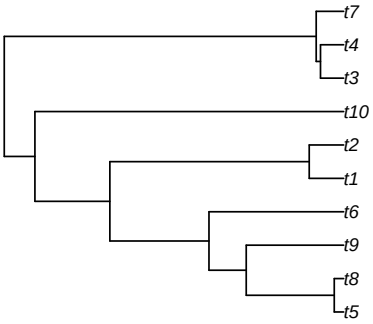
Question (ii)

```
length.phylo = function(phylo) {
  sum(phylo$edge.length)
}
```

Question (iii)

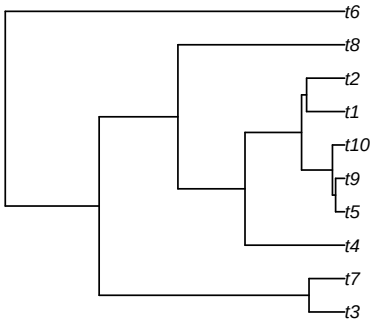
```
n = 10
lambda =  0.5
par(mfrow=c(2,2))

for (i in 1:4) {
  phylo = yaPhylo(n, lambda)
  plot(phylo)
  title(main=paste("Tree", i),
        sub=paste("Phylogenetic Diversity:", round(length(phylo),2)))
}
```

Tree 1



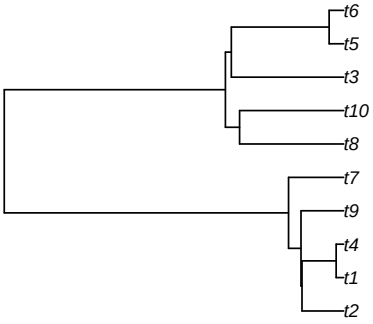
Phylogenetic Diversity: 6.68

Tree 2



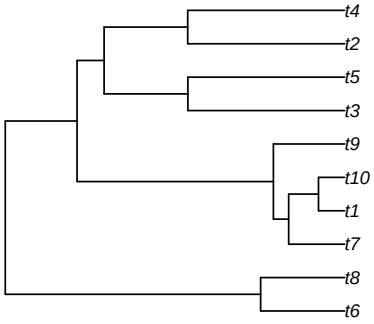
Phylogenetic Diversity: 36.98

Tree 3



Phylogenetic Diversity: 11.09

Tree 4



Phylogenetic Diversity: 7.77

Part 3