Appendix 1

Brian O’Meara & Jeremy Beaulieu

6/30/2020

## Functionally congruent coin models

Coin flipping models are popular examples to use for statistics. Here we create models that imply very different things about coins, but have indistinguishable likelihoods at the precision commonly used in phylogenetics (hundredths of log likelihood units).

First start with the simple binomial model:

coin\_simple\_likelihood <- function(x, nheads, nflips) {  
 return(dbinom(nheads, nflips, x[1]))   
}

However, we can also have models that give a different probability of heads with every flip changing smoothly according to a function. Maybe we wear down the face of the coin every time we flip it so it is less and less likely to land on the lighter face.

coin\_linear\_prob <- function(x, fixed.slope=0.1, nheads, nflips) {  
 probabilities <- fixed.slope\*sequence(nflips)/nflips + x[1]  
 return(probabilities)  
}

We can make several different models with different fixed.slope: one increases the probability of heads with every flip, another decreases, for example. But for each we can fit x[1], the probability of heads before any flips.

Since the probability of heads can change with every flip, calculating the probability of an observed proportion of heads in a set of flips requires computing the probability of all ways to get there: 2 heads of 3 flips could be HHT, HTH, THH, and each of these has a different probability.

coin\_linear\_likelihood <- function(x, nheads, nflips, fixed.slope=0.1, neg=FALSE) {  
 probabilities <- coin\_linear\_prob (x=x, nheads=nheads, nflips=nflips, fixed.slope=fixed.slope)  
 if(any(probabilities>1 | probabilities<0)) {  
 return(-1e8)  
 }  
 totalheadsprobs <- rep(0, nflips+1) #so it goes from 0 to nflips total heads  
 for (i in seq\_along(probabilities)){  
 probhead <- probabilities[i]  
 probtail <- 1-probhead  
 if(i==1) {  
 totalheadsprobs[0+1] <- probtail  
 totalheadsprobs[1+1] <- probhead  
 } else {  
 totalheadsprobs <- totalheadsprobs\*probtail + c(0,totalheadsprobs[1:(length(totalheadsprobs)-1)]\*probhead)  
 }  
 }  
 return(ifelse(neg, -1, 1)\*totalheadsprobs[nheads+1])  
}  
  
coin\_exponential\_prob <- function(halflife, nheads, nflips) {  
 lambda <- log(2)/halflife  
 pheads <- 1\*exp(-lambda\*seq(from=1, to=nflips, by=1)) #by not starting at exp(-lambda \* 0), it allows for something that has no heads on the first flip to have positive probability  
 return(pheads)  
}  
  
get\_possibilities <- function(heads, flips) {  
 if(flips>25) {  
 stop("Too many flips for the expand.grid silly approach to work")  
 }   
 possibilities <- expand.grid(rep(list(c(0,1)),flips))  
 possibilities <- possibilities[rowSums(possibilities)==heads,]  
 return(possibilities)  
}  
  
coin\_exponential\_likelihood <- function(halflife, nheads, nflips, neg=FALSE, possibilities=get\_possibilities(heads=nheads, flips=nflips)) {  
   
 pheads <- coin\_exponential\_prob(halflife, nheads, nflips)  
 ptails <- 1-pheads  
 prob\_matrix <- rbind(ptails, pheads) # heads = 1, tails =0, so when we offset, tails is first row, heads second  
   
 likelihood <- 0  
 for (i in sequence(nrow(possibilities))) {  
 positions <- as.numeric(possibilities[i,]+1)  
 local\_prob <- 1  
 for (j in sequence(ncol(possibilities))) {  
 local\_prob <- local\_prob \* unname(prob\_matrix[positions[j],j])  
 }  
 likelihood <- likelihood + local\_prob  
 }  
 print(c(halflife, likelihood))  
 return(ifelse(neg, -1, 1)\*likelihood)  
}

Now let’s simulate some data:

nflips <- 100  
pheads <- 0.3  
nheads <- rbinom(n=1, size=nflips, prob=pheads)

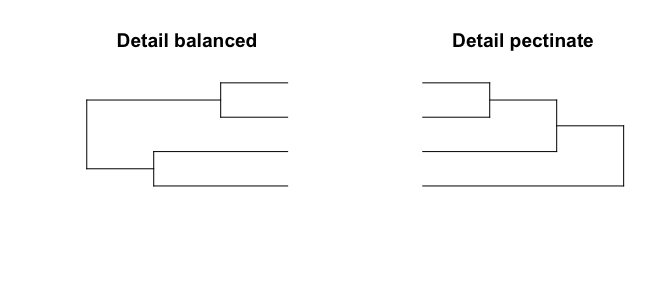
And fit it to three models: one with a constant estimated probability of heads, one with a linear increase in probability of heads with each flip (so by the end the probability of heads is 10% higher than when it started)

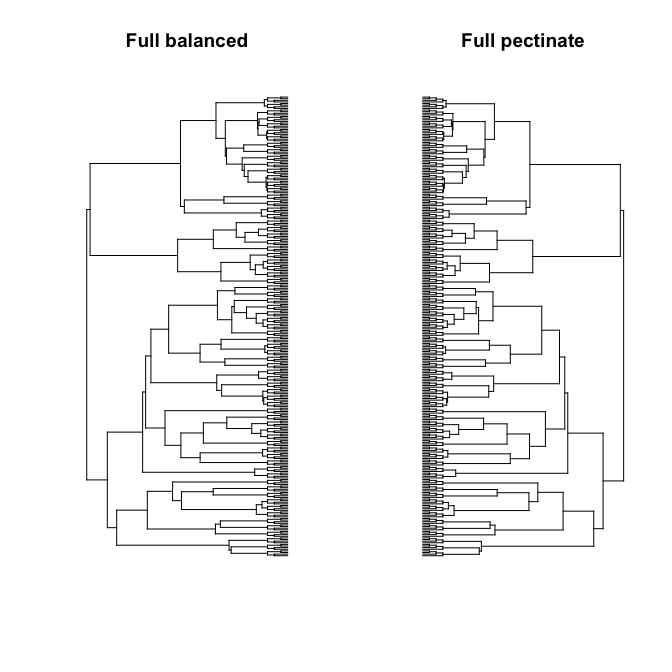
And we can plot the results

## Diversification Signal

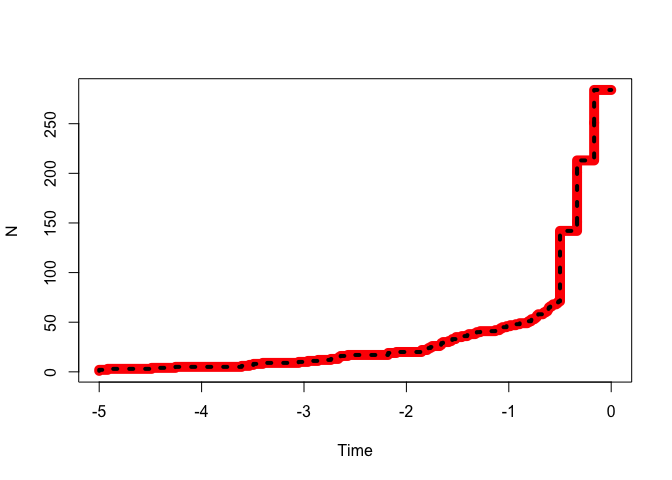
An open question in Louca and Pennell (2020) is the status of SSE models, such as BiSSE, MuSSE, HiSSE, and more. To examine this, we computed two trees with equal lineage through time plots. They have the same backbone tree, but then one replaces every tip with a balanced tree of four taxa ((A,B),(C,D)), and one uses a pectinate tree (A,(B,(C,D))).

## b d   
## 1.6838698 0.4839536





And the lineage through time plots are the same for both trees (red and black lines, respectively)



We can fit a few models to these trees

## Initializing...   
## Finished. Beginning simulated annealing...   
## Finished. Refining using subplex routine...   
## Finished. Summarizing results...

## Initializing...   
## Finished. Beginning simulated annealing...   
## Finished. Refining using subplex routine...   
## Finished. Summarizing results...

## Initializing...   
## Finished. Beginning simulated annealing...   
## Finished. Refining using subplex routine...   
## Finished. Summarizing results...

## Initializing...   
## Finished. Beginning simulated annealing...   
## Finished. Refining using subplex routine...   
## Finished. Summarizing results...

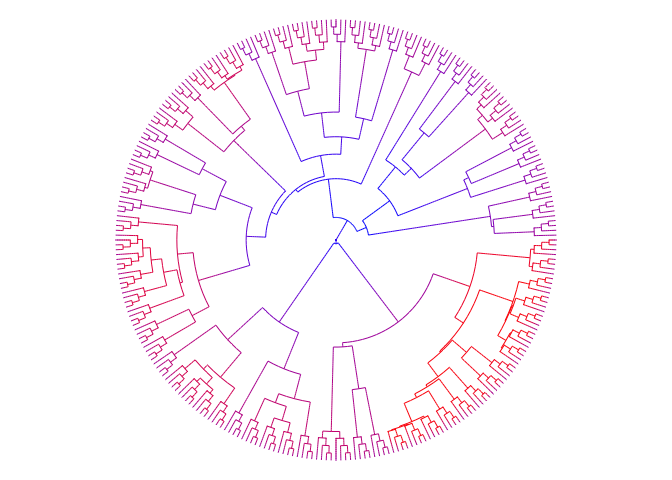
|  |  |  |
| --- | --- | --- |
| model | balanced | pectinate |
| Yule | -175.6911 | -175.6911 |
| BD | -174.2795 | -174.2795 |
| MiSSE 1 | -174.2795 | -174.2795 |
| MiSSE 2 | -172.2606 | -172.2433 |

## Calculating marginal probabilities for 283 internal nodes...   
## Finished. Calculating marginal probabilities for 284 tips...   
## Done.

## Calculating marginal probabilities for 283 internal nodes...   
## Finished. Calculating marginal probabilities for 284 tips...   
## Done.



## $rate.tree  
## Object of class "contMap" containing:  
##   
## (1) A phylogenetic tree with 284 tips and 283 internal nodes.  
##   
## (2) A mapped continuous trait on the range (0.770867, 1.6824).



## $rate.tree  
## Object of class "contMap" containing:  
##   
## (1) A phylogenetic tree with 284 tips and 283 internal nodes.  
##   
## (2) A mapped continuous trait on the range (0.769287, 1.684501).