Correlation models

Brian C. O'Meara

14 January, 2020

You can do this on your own data, or on included data here.

##Continuous data

```
library(geiger)
library(pic)
tree.primates <- read.tree(text="((((Homo:0.21,Pongo:0.21):0.28,Macaca:0.49):0.13,Ateles:0.62):0.38,Gala
X <- c(4.09434, 3.61092, 2.37024, 2.02815, -1.46968)
Y <- c(4.74493, 3.33220, 3.36730, 2.89037, 2.30259)
names(X) <- names(Y) <- c("Homo", "Pongo", "Macaca", "Ateles", "Galago")
pic.X <- pic(X, tree.primates)
pic.Y <- pic(Y, tree.primates)
```

Now, positivitize the contrasts and do a regression through the origin.

##Discrete data

```
require("corHMM")
?corHMM
data(primates)
ls()
print(primates)
require(phytools)
```

Just to make this a better dataset for our exercise, I'm going to change some of the states (I want to have all four trait combinations present). For actual analyses, of course, DO NOT INVENT YOUR DATA.

First, a review of discrete state models:

```
primates$trait[which(grepl("Hylobates",primates$trait[,1])),2]<-1

trait1<-primates$trait[,2]
names(trait1)<-primates$trait[,1]
plotSimmap(make.simmap(primates$tree, trait1), pts=FALSE, fsize=0.8)
rate.mat.er<-rate.mat.maker(rate.cat=1, hrm=FALSE, ntraits=1, nstates=2, model="ER")
print(rate.mat.er)
```

What does this matrix mean?

```
pp.er<-corHMM(primates$tree,primates$trait[,c(1,2)],rate.cat=1,rate.mat=rate.mat.er,node.states="margina
print(pp.er)
```

What do these results mean?

```
rate.mat.ard<-rate.mat.maker(rate.cat=1, hrm=FALSE, ntraits=1, nstates=2, model="ARD")
print(rate.mat.ard)
```

And these?

```
pp.ard<-corHMM(primates$tree,primates$trait[,c(1,2)],rate.cat=1,rate.mat=rate.mat.ard,node.states="marg
print(pp.ard)
```

which model is better?

Now let's look at multiple traits.

This is a matrix with four states

```
rate.mat.er.4state<-rate.mat.maker(rate.cat=1, hrm=FALSE, ntraits=1, nstates=4, model="ER")
print(rate.mat.er.4state)
```

Convert the two binary traits into a single four character state

```
fourstate.trait<-rep(NA,Ntip(primates$tree))
for(i in sequence(Ntip(primates$tree))) {
    if(primates$trait[i,2]==0 && primates$trait[i,3]==0) {
        fourstate.trait[i]<-0
    }
    if(primates$trait[i,2]==0 && primates$trait[i,3]==1) {
        fourstate.trait[i]<-1
    }
    if(primates$trait[i,2]==1 && primates$trait[i,3]==0) {
        fourstate.trait[i]<-2
    }
    if(primates$trait[i,2]==1 && primates$trait[i,3]==1) {
        fourstate.trait[i]<-3
    }
}
fourstate.data<-data.frame(Genus_sp=primates$trait[,1], T1=fourstate.trait)

print(rayDISC(primates$tree, fourstate.data, ntraits=1, model="ER", node.states="marginal"))
print(rayDISC(primates$tree, fourstate.data, ntraits=1, rate.mat=rate.mat.er.4state, node.states="margi
rate.mat.ard.4state<-rate.mat.maker(rate.cat=1, hrm=FALSE, ntraits=1, nstates=4, model="ARD")
print(rate.mat.ard.4state)
```

Now let's make the equivalent of a GTR matrix:

```
rate.mat.gtr.4state<-rate.mat.ard.4state
rate.mat.gtr.4state<-rate.par.eq(rate.mat.gtr.4state, c(1,4))
rate.mat.gtr.4state<-rate.par.eq(rate.mat.gtr.4state, c(2,6))
rate.mat.gtr.4state<-rate.par.eq(rate.mat.gtr.4state, c(3,8))
rate.mat.gtr.4state<-rate.par.eq(rate.mat.gtr.4state, c(4,6))
rate.mat.gtr.4state<-rate.par.eq(rate.mat.gtr.4state, c(5,7))
rate.mat.gtr.4state<-rate.par.eq(rate.mat.gtr.4state, c(6,7))
print(rate.mat.gtr.4state)

print(rayDISC(primates$tree, fourstate.data, ntraits=1, rate.mat= rate.mat.gtr.4state, node.states="marg
```

Now make a model like Pagel 1994

```
print(rate.mat.maker(rate.cat=1, hrm=FALSE, ntraits=2, nstates=2, model="ARD"))
rate.mat.pag94<-rate.par.drop(rate.mat.ard.4state, drop.par=c(3,5,8,10))
print
```

Now that you have some introduction, there are two routes:

##Route 1

Construct a model to test if state 1 can never be lost

Experiment with the effects of frequencies at the root.

Create and use a model to see if transitions from 00 go to 11 only via 01.

##Route 2

Maddison and FitzJohn (2015) pretty convincingly show (to me) that Pagel (1994) is just not a good method. Ok. So work on a fix. They point to Read and Nee (1995) as a low power but possible solution. Look at their appendix, especially, and write an implementation.