hOUwie notes

Jeremy M. Beaulieu

Background

The goal of this project is to design a framework for simultaneously optimizing a discrete character model while also fitting a continuous character model wit OUwie. The idea behind this is that we assume a correlation between the two trait types, but we reconstruct the regimes first, then fit OUwie separately. This is not what we want, because change on a branch may not be optimal without information from how the regimes evolved; where regimes change may not be optimal without information from how rates change. This approach should not be confused with the "hypothesis-free" approaches like SURFACE or AUTEUR. Such models are useful, though in my view, the stronger the α, the more difficult it should become to reconstructe regimes correctly. Moreover, with methods like Brownie and OUwie, we often go into these analyses with an *a priori* hypothesis about what might be governing the rates and optima. So hOUwie is intended to meet everyone halfway in this regard, as well as to fix an issue that seems to have gone unnoticed as to how these analyses are conducted (though see Revell 2013). Also, it's worth pointing out that by connecting hidden Markov models to identifying regimes in OUwie would bring discovery into these types of model fitting exercises.

My first attempt at this was as follows: with the joint reconstruction find the likeliest state for each node in the tree using the algorithm of Pupko et al (2000). To obtain the true likelihood of a reconstruction, I needed to assess changes along a branch as well as a node. To do so was actually simple: add in a large set of nodes of degree 2 down a branch. Unfortunately, there is this strange property where change occurs either at the instant of speciation or right before, which depended on the asymmetry of the transition rates.

The next step is to try a hierarchical maximum likelihood approach using the stochastic mapping now implemented in `corHMM()`. I will outline the equation as well as a skeleton code function for how this can be accomplished below.

Linking corHMM and OUwie

The likelihood function is computed by summing over all possible maps.

$$L(X, D|\Theta, \Phi) = P(D|\Phi) \sum_{z} P(z|D, \Phi) g(X|\Theta, z)$$

where $P(D|\Phi)$ is the probability of observing the vector states, D, at the tips of a phylogeny given parameters Φ, and $P(z|D, \Phi)$ provides the probability of a character map z given a vector of character state D and parameters Φ (note I am using Φ to generically define any model that will estimate a character history). The mixture probability, $g(X|\Theta)$, is the BM or OU likelihood given a set of parameters Θ and character map z.

It is unclear how one would go about calculating $P(z|D, \Phi)$, especially with a larger tree and many character states. Instead, it is easier to sample a large number of realizations of the character transitions given D and Φ. This is where the stochastic maps come in. The summation above would then be over a set of unique character maps that appear in the sample. The frequency of each unique map, f_z, would then be used as an estimate of $P(z|D, \Phi)$. If each character map is unique, then the summation is simple an average of likelihoods across the set of maps. This approach is costly from a computational perspective because for every new set of Φ proposed a new set of character maps would have to be generated and reevaluated. It is also unclear at the moment how many maps would suffice to get a good approximation of $P(z|D, \Phi)$, but this will be something we will have to work out in simulation.

Considerations

1. How many maps per iteration of Φ? This part will be computationally expensive.
2. What is the appropriate null model for something like this? It explictly defines the character and the continuous trait as being correlated. However, assuming the null likelihood as being logLik(OU1) + logLik(corHMM) seems far too simplistic. This is something that will require some thought.
3. I don't like the idea of simply fitting OUM then try OUMV, then OUMA, then OUMVA, and then compare using AIC. I think a better solution is following how parameters are estimated in the new `OUwie.dredge()` function, which is a "find me my hypotheses" algorithm. In this function users choose, at most, how many sigma and alpha parameters there can be, and the algorithm toggles between these max settings and simpler combinations. What you end up with, for example, is the possibility of having 3 regimes, 1 sigma, 2 alphas and a single theta. I think this sort of thing is what we want here. The question is whether just using likelihood is best or whether AIC or BIC type thing is best here, given that more complex will always lead to better likelihood.