

Utilizing **maticce** to estimate transitions in continuous character evolution

Andrew Hipp

January 14, 2009

1 Introduction

This document provides an overview of the **maticce** package, which serves three primary purposes. First, it implements an information-theoretic approach to estimating where on a phylogeny there has been a transition in a continuous character. As currently implemented, the approach assumes that (1) such transitions are appropriately modeled as shifts in optimum / equilibrium of a character evolving according to an Ornstein-Uhlenbeck process; (2) strength of constraint / rate of evolution toward the optimum is constant over the tree, as is variance; and (3) all branches on which a change could occur are identified. These assumptions can be relaxed in future versions if needed. Second, the package provides helper functions for the **ouch** package, in which all likelihood calculations are performed. For example, the package automates the process of painting regimes (described in the *Painting Regimes* section below) for the **hansen** function of **ouch**, specifying nodes at which the regime changes. It also provides functions for identifying most recent common ancestors and all descendents of a particular node. Users of **ouch** who want to handle large numbers of analyses may find the routines for summarizing analyses over trees and over regimes useful as well. Finally, **maticce** provides a flexible set of simulation functions for visualizing how different model parameters affect (i.e., what they 'say' about) our inference of the evolution of a continuous character on a phylogenetic tree.

This document also provides a worked example of analyzing a continuous character dataset that illustrates most of the **maticce** features. Working through this example will I expect address most questions that should come up during a typical analysis.

2 Package Overview

The **maticce** package currently implements functions in the following categories:

- Extracting information from **ouch**-style (S4 class **ouchtree**) trees
- Painting regimes on a batch of **ouch**-style trees

- Performing batch analyses in `ouch` over trees and over regimes
- Summarizing analyses
- Simulating data under explicit models for visualization purposes

Functions in the first two categories will be of general utility to `ouch` users; functions in the third category are utilized to perform analyses over trees and over models, and in the fourth to summarize these with regard to the hypothesis that there has been a shift in continuous character on a given branch of a phylogenetic tree. While these latter two categories are more specific, the functions can be easily modified to address other multiple-tree / multiple-model questions.

3 Getting started

In case you aren't familiar with R, the following commands will get you started.

```
> #library(maticce) # load the maticce package into memory; also loads the required ouch packages
> library(ouch)
> data(carex)
> attach(carex)
> ovales.tree <- ape2ouch(ovales.tree) # make the tree an ouchtree object
```

Note that although the sample tree provided (`carex[['ovales.tree']]`) is ultrametric, this is not strictly required.

4 Extracting information from trees

Three functions are available to extract information from an `ouchtree` object:

- `isMonophyletic`: returns a T or F depending on whether the taxa identified are monophyletic on the tree provided
- `nodeDescendants`: identifies all descendants of a given node on a tree
- `mrcaOUCH`: identifies the most recent common ancestor of a given set of taxa

These functions can be used interactively to identify nodes on the tree for analysis. Because the batch-analysis functions in `maticce` identify nodes based on taxa (see explanation in the section on 'Performing batch analyses'), nodes are provided as a list of vectors, each vector containing all taxa descendent from the node of interest. You can generate these lists manually by typing lists of names into vectors, or you could use the following if you want to explicitly designate all taxa for each node by selecting from a list:

```
> nodes <- list(8) # assuming you want 8 nodes}
> for(i in 1:length(nodes)) nodes[[i]] <- select.list(otree@nodelabels, multiple = T)}
```

Alternatively, if you want to designate the node more quickly by just selecting the most recent common ancestor of a set of taxa:

```
> for(i in 1:length(nodes)) {
>   ancestor <- mrcaOUCH(select.list(otree@nodelabels, multiple = T), otree)
>   nodes[[i]] <- nodeDescendants(otree, ancestor)
> }
```

These functions are all documented under `isMonophyletic`.

5 Painting regimes

In the `hansen` function of `ouch`, Ornstein-Uhlenbeck models are specified by specifying for each phylogenetic branch one and only one selective regime that governs the evolution of individuals that occupy that branch. In the `maticce` approach, *selective regime* is an overly specific description, because the dynamics of trait evolution may shift significantly at cladogenesis for reasons that have nothing to do with natural selection. For consistency with `ouch`, the term *regime* is retained in `maticce`, but it is used here to refer to the entire set of lineage-specific stationary distributions on a tree rather than the branch-specific set of selective pressures that is implied by *selective regime*. Hereafter, and in the `maticce` documentation, *regime* is used interchangeably for the tree-based model (the vector returned by `paintBranches` and visualized using `plot(tree, regimes=regime)`). Two functions are available for painting regimes; both return objects that may be used directly in the `hansen` function of `ouch`:

- `paintBranches`: returns the single regime for character transitions occurring at all specified nodes
- `regimeVectors`: returns all possible regimes for specified nodes, up to a maximum of `maxNodes` transitions
- `regimeMaker`: returns regimes defined by a matrix, with each row specifying which nodes permit character transitions

The `paintBranches` function is typically called from within `regimeVectors`, but it can be called separately. Nodes can be designated by number or taxa; the function assumes the latter only if it receives a list to evaluate instead of a vector.

```
> ou2 <- paintBranches(list(ovales.nodes[[2]]), ovalues.tree)
```

The regime can be used directly in a call to `hansen` or the `plot` method for an `ouchtree` object (Figure 1). Note that `paintBranches` paints the crown

```
> plot(ovales.tree, regimes = ou2, cex = 1.2)
```

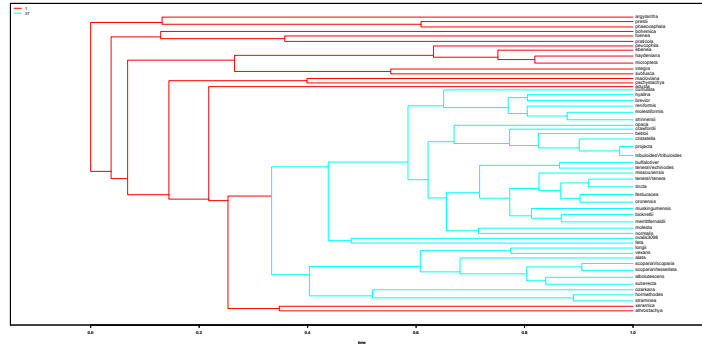


Figure 1: `ovales.tree` with coloring according to `ou2`

group designated by the taxa you give it. As written now, there is not an option to begin painting on the branch above that node (i.e., to paint the stem groups designated by your list of taxon-vectors). In practice, this is not likely to affect your conclusions. However, it might, because the Ornstein-Uhlenbeck calculations integrate over (1) the amount of time that a lineage occupies each component of the regime and (2) the amount of time elapsed since the end of each regime component. If this is important to you, write me, and we can adjust the `paintBranches` function to allow a mix of branch-based and node-based regime definitions.

6 Batch analyses

The goal of `maticce` is to make regime-definition and batch analyses of multiple models and multiple trees straightforward, so that researchers can focus on specifying their models and interpreting the results rather than on the book-keeping of running numerous analyses. The things a researcher should be thinking about are:

- *Which nodes are you interested in testing?* The choice of which nodes you are considering will have the strongest effect on your estimates of the support for a character transition having occurred at those nodes. This is a standard issue in model-fitting: the choice of which models to consider is the primary question once you have data in hand.
- *How many transitions are plausible on a single tree?* The feasibility of studying a large number of nodes is governed by how many simultaneous transitions you allow. Suppose you have 15 nodes that are of interest. Testing models that allow transitions at anywhere between zero and 15 nodes would entail testing 32,768 models. This would be too long to be

feasible. Allowing changes at anywhere between zero and four nodes would entail testing a more manageable 1,941 models.

- *How much do you trust poorly-supported nodes? Do you want to consider them at all?* `maticce` allows you to analyze over a set of trees, e.g. trees visited in a Bayesian (MCMC) analysis or a set of bootstrap trees. The `summary` function will give you an estimate of the support for a transition at each node you specify, both conditioned on trees that possess that node and averaged over all trees. If you have some reason for trusting the node in spite of low support (because, for example, it holds together a morphologically coherent group), you might want to give some credence to the support value that conditions only on trees that possess that node.

```
> # First, analyze with maxNodes set to 2
> ha.4.2 <- runBatchHansen(ovales.tree, ovalues.data, ovalues.nodes[1:4], maxNodes = 2, brown
> print(summary(ha.4.2))
```

Summarizing hansenBatch analyses over 1 trees and 12 models

ESTIMATED SUPPORT FOR CHANGES OCCURRING AT DESIGNATED NODES

Averaged over all trees:

	1	2	3	4
AIC.weight	0.4306845	0.8848683	0.15490978	0.15202585
AICc.weight	0.4060012	0.8824765	0.14168053	0.13901638
BIC.weight	0.3189039	0.8740303	0.09500061	0.09311196

Support conditioned on trees that possess the node

	1	2	3	4
AIC.weight	0.4306845	0.8848683	0.15490978	0.15202585
AICc.weight	0.4060012	0.8824765	0.14168053	0.13901638
BIC.weight	0.3189039	0.8740303	0.09500061	0.09311196

MODEL-AVERAGED PARAMETERS

alpha = 312.4266

sigma^2 = 5.192254

```
> # Then, analyze with maxNodes set to 4
> ha.4.4 <- runBatchHansen(ovales.tree, ovalues.data, ovalues.nodes[1:4], maxNodes = 4, brown
> print(summary(ha.4.4))
```

Summarizing hansenBatch analyses over 1 trees and 17 models

ESTIMATED SUPPORT FOR CHANGES OCCURRING AT DESIGNATED NODES

Averaged over all trees:

	1	2	3	4
AIC.weight	0.5426478	0.9039240	0.2877716	0.2909109
AICc.weight	0.4914288	0.8965165	0.2385572	0.2404165

```
BIC.weight  0.3551409 0.8795015 0.1317644 0.1315286
```

```
Support conditioned on trees that possess the node
              1          2          3          4
AIC.weight  0.5426478 0.9039240 0.2877716 0.2909109
AICc.weight 0.4914288 0.8965165 0.2385572 0.2404165
BIC.weight  0.3551409 0.8795015 0.1317644 0.1315286
```

```
MODEL-AVERAGED PARAMETERS
```

```
alpha = 316.4656
sigma^2 = 5.246654
```

```
> ouSim.ha.4.2 <- ouSim(ha.4.2, tree = ovales.tree)
> plot(ouSim.ha.4.2, colors = ou2)
```

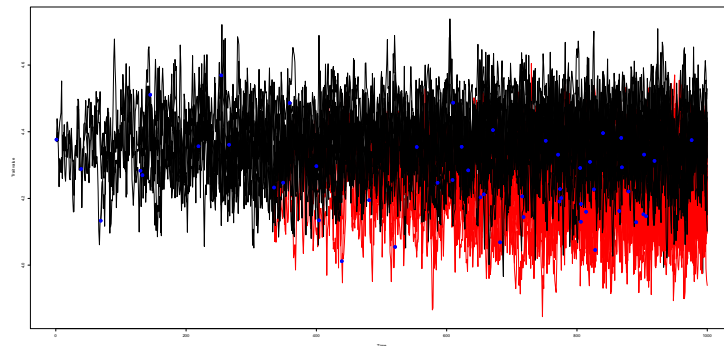


Figure 2: Simulated character on `ovales.tree` at model-averaged theta values, with coloring according to `ou2`

What is the relative support for the brownian motion vs. OU-2 model? We find which model has a change only at node 2 by inspecting the regime matrix

```
> ha.4.2[['regMatrix']][['overall']]
```

```
      1 2 3 4
1  1 1 0 0
2  1 0 1 0
3  1 0 0 1
4  1 0 0 0
5  0 1 1 0
6  0 1 0 1
7  0 1 0 0
8  0 0 1 1
9  0 0 1 0
```

```
10 0 0 0 1
11 0 0 0 0
```

Then we can find the likelihood of these models on a given tree:

```
> ha.4.2[['hansens']][[1]]
```

	loglik	dof	sigma.squared	theta / alpha
1	52.72744	5	5.31005363	331.626872
2	50.54000	5	5.86424413	337.252821
3	50.21561	5	6.04474990	343.369492
4	49.95720	4	5.14354118	289.387746
5	51.75330	5	5.12434568	308.473054
6	51.81077	5	4.84664447	292.440242
7	51.75142	4	5.16258108	310.762680
8	44.31518	5	6.55448190	297.993592
9	44.17736	4	8.49647580	384.329498
10	42.89766	4	6.87292796	296.238909
11	39.57676	3	0.09810907	3.341806
brown	31.50503	2	0.04205436	4.351195

or the information criterion weights:

```
> summary(ha.4.2)[['modelsMatrix']][[1]]
```

	AICwi	AICcwi	BICwi
1	3.158399e-01	2.887904e-01	1.933452e-01
2	3.543867e-02	3.240359e-02	2.169421e-02
3	2.562102e-02	2.342676e-02	1.568422e-02
4	5.378499e-02	6.138046e-02	8.818021e-02
5	1.192348e-01	1.090232e-01	7.299104e-02
6	1.262885e-01	1.154727e-01	7.730903e-02
7	3.235051e-01	3.691902e-01	5.303850e-01
8	7.015685e-05	6.414841e-05	4.294738e-05
9	1.661543e-04	1.896184e-04	2.724090e-04
10	4.621088e-05	5.273673e-05	7.576248e-05
11	4.537212e-06	6.148318e-06	1.992244e-05
brown	3.851020e-09	5.912695e-09	4.528691e-08