An introduction to the EcoPhyl package

Matt Helmus (???) and Will Pearse (wdpearse@umn.edu)

November 2013

Contents

1 Installing EcoPhyl

You can install EcoPhyl by typing `install.packages("EcoPhyl", dependencies=TRUE)`, and get a listing of the functions in the package by typing `library(help=EcoPhyl)`. When loading EcoPhyl into your R session (`library(EcoPhyl)`), you may notice a warning about a package named 'ecoPD'. ecoPD is not maintained on CRAN like many other packages, but can still be installed by typing install.packages("ecoPD", repos="http://R-For

If you find any bugs, or have any feature requests for the package, please use the online tracker at `http://github.com/willpearse/XXX/issues`. Indeed, all the

package's code is available at `http://github.com/willpearse/XXX/issues`, where you can suggest modifications and alterations as you wish! EcoPhyl is an evolving package; new functions and features will be added as we publish papers describing them.

Please make sure you cite the accompanying paper for EcoPhyl (XXX) when publishing work using it. Also, while `EcoPhyl` contains much novel code, it relies heavily on the *R* ecosystem. Much of the community phylogenetic metric functions are wrappers around existing code (detailed in the help files for each function), in particular code from `picante`, `caper`, and `ecoPD` (REFS XXX)—please cite the authors of these packages so that their hard-work is rewarded!

2 Data formats in EcoPhyl

EcoPhyl functions work with 'comparative community ecology' objects, called `comparative.comm` objects. These are designed to help keep phylogenies, community data matrices, species trait data, and environmental data all in the same place in a format that makes it easy to work with them. They're much less scary than they sound!

Below we load EcoPhyl, some example data that comes with it, and then make a comparative.comm object. You can examine the phylogeny ('tree'), community data ('comm'), and trait data ('traits') that went into making dataset for yourself, although all the data types are explained in more detail below.

```
> library(EcoPhyl)
> data(EcoPhyl)
> data <- comparative.comm(traits.and.phy$phy, traits.and.phy$comm, traits.and.phy$
> data

Comparative community dataset of 40 taxa:
Phylogeny: traits.and.phy$phy
   39 internal nodes, VCV matrix present
Community data: traits.and.phy$comm
     50  sites
Trait data: traits.and.phy$traits
    10  variablesEnvironmental data: None
```

EcoPhyl is conservative; if you give it trait data for only half of the species in your community data, the comparative.comm object will only contain data on

those species that have both trait data and community data. The same goes for the phylogeny, and for sites with environmental data. EcoPhyl will warn you about the loss of species or traits when you print the object to screen, and while it's making the comparative.comm object (unless you set the argument warm=FALSE).

You can also subset your comparative.comm object to exclude certain species or sites, in much the same way you can a data.frame. Note that EcoPhyl will not (by default) warn you if this operation drops out certain species or sites. For example:

```
> data[1:5,]

Comparative community dataset of 40 taxa:
Phylogeny: x$phy
   39 internal nodes, VCV matrix present
Community data: comm
     5  sites
Trait data: new.x$traits
    10  variablesEnvironmental data: None

> data[,1:3]

Comparative community dataset of 3 taxa:
Phylogeny: phy
   2 internal nodes, VCV matrix present
Community data: comm
     50  sites
Trait data: traits
    10  variablesEnvironmental data: None

> data[,1:3,warn=TRUE]

Comparative community dataset of 3 taxa:
Phylogeny: phy
   2 internal nodes, VCV matrix present
Community data: comm
     50  sites
Trait data: traits
    10  variablesEnvironmental data: None
```

2.1 Phylogenies

EcoPhyl uses the `phylo` format in the `ape` package to store phylogenues. You can load your own phylogenies using the ape functions 'read.tree' and 'read.nexus'.

2.2 Community data

EcoPhyl uses the same community data format as the `vegan` package: a matrix or data.frame with sites in the rows and species in the columns. The elements of the community matrix can be species abundances or presence/absence (1/0). Not all the species in your matrix have to be present in a site, i.e. there can be empty columns in your data. This is particularly important when using the dispersion measures (see below). Your data should be named, with row names have correspond to sites, and column names that correspond to species.

2.3 Trait data

Trait data should be a data.frame with row names that correspond to the species in the phylogeny, and named columns for each separate trait.

2.4 Environemntal data

Environmental data should be a data.frame with row names that correpsond to the sites in your community data, and separate (named) columns for each kind of environmental data.

3 Plotting and exploring data

EcoPhyl comes with a few functions that are intended to make exploring your data slightly easier. For instance, you can plot out graphs of species abundances in communities

```
> cc.dotplot(data)
> #cc.barplot(data, c(""))
```

4 Community phylogenetic metrics

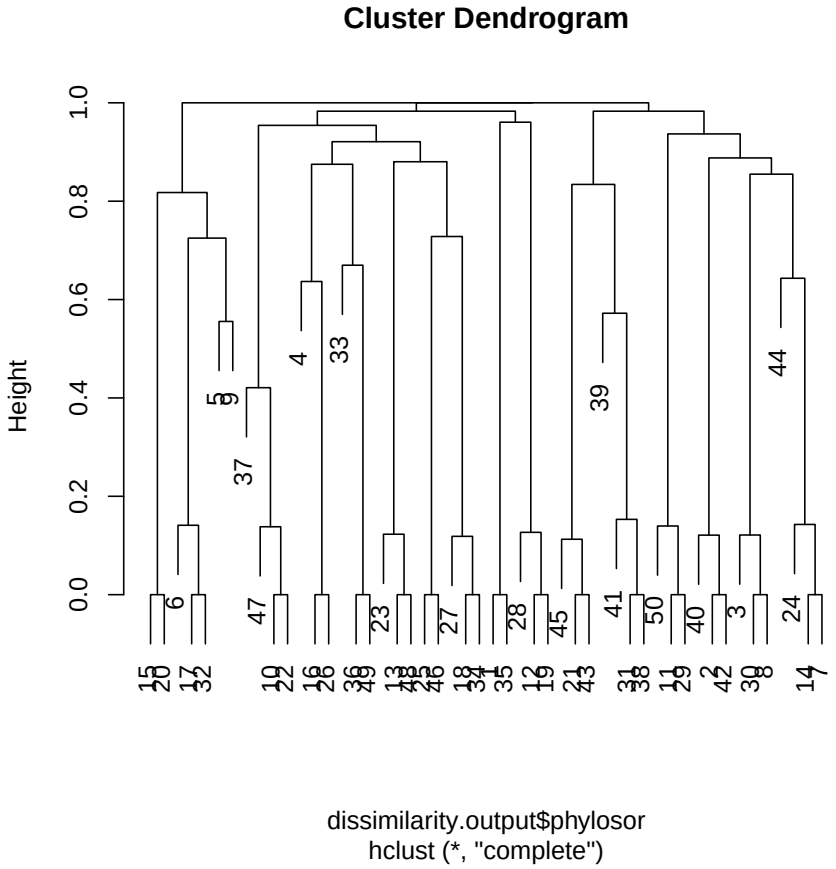
EcoPhyl splits community phylogenetic metrics into four functions according to the scheme outlined by Pearse et al. (XXX): `shape`, `evenness`, `dispersion`, and `dissimilarity`. In brief, *shape* metrics ignore abundances, *evenness* metrics incorporates abundances, *dispersion* metrics compare observed communities with source pools of potential species, and *dissimilarity* metrics compare one community with another.

You can calculate all metrics within a class at the same time (which is what we recommend), or you can pick a particular one. Below we show how to calculate the metrics, and give examples of how to work with their output.

```
> shape.output <- shape(data)
> shape.output
```

```
Shape metrics in this object:
        PSV (psv)
        PSR (psr)
        Mean Phylogenetic Distance (mpd)
        Phylogenetic Distance (pd)
        Phylogenetic Distance stanardised according to number of species in sample (p
        Colless' index (colless)
        Gamma (gamma)
        Taxonomic diversity (delta)
        Sum of dominant eigenvector (eigen.sum)
        Cadotte's expected phylogenetic diversity (cadotte.pd)
Use something like 'output$psv' to work with each measure
```

```
> shape.output$mpd

 [1] 15.054258 19.074699 19.084836 20.062390 14.727638 17.056183
 [7] 15.639555 16.472732 16.758934 21.582280 16.839068 13.228265
[13] 16.428817 24.747868 15.639555 19.577438 18.313019 19.887967
[19] 12.864086 17.730220 15.506598 15.639555 17.730220 25.078200
[25]  9.295193  9.295193 14.680021 16.756560 18.942210 16.152897
[31] 24.063442 15.042695 19.887967 18.828898 21.414410 17.097972
[37] 19.441666 15.054258 18.828898 18.571968  9.295193 17.056463
[43] 24.260742 13.228265 14.593654 16.626518 17.452182 13.228265
[49] 16.756560 18.178656

> dissimilarity.output <- dissimilarity(data, metric="phylosor")
> plot(hclust(dissimilarity.output$phylosor))
```



5 Eco-evolutionary regression (EcoPhyl)

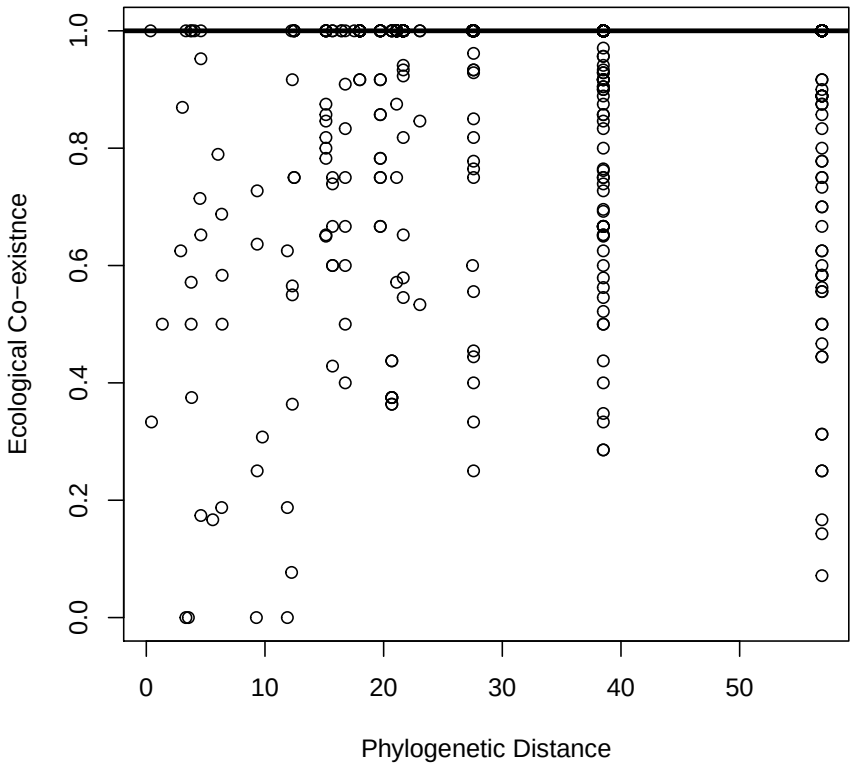
EcoPhyl is intended to replace and improve upon earlier Visual Basic/C programs called EcoPhyl. You can regress the relative coexistence of species in your dataset against those species phylogenetic (`eco.phy.regression`) and trait (`eco.trait.regression`) dissimilarity, as well as shared habitat preferences based on environmental tolerances (`eco.env.regression`). Note that you can examine all traits at the same time, or each trait separately. Below are some examples.

```
> model <- eco.trait.regression(data, method="lm")
> model

eco.trait.regression regression
Method:  lm
```

```
Randomisation: NONE
Observed slope:  0.04
Observed model summary:

Call:
lm(formula = as.numeric(eco.mat) ~ as.numeric(trait.mat))

Residuals:
     Min       1Q   Median       3Q      Max
-0.84725  0.00557  0.05574  0.10550  0.24915

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)          0.724104   0.024011  30.157  < 2e-16 ***
as.numeric(trait.mat) 0.043370   0.005369   8.078 2.51e-15 ***
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

Residual standard error: 0.187 on 778 degrees of freedom
Multiple R-squared:  0.07738,        Adjusted R-squared:  0.07619
F-statistic: 65.25 on 1 and 778 DF,  p-value: 2.508e-15

> more.complex.model <- eco.trait.regression(data, method="mantel", altogether=FALS
> more.complex.model

eco.trait.regression list:
Traits:
A, B, C, D, E, F, G, H, I, J
To examine each regression of each trait, use something like 'x[[1]]', or 'print(x[[2
To display all at once, call something like 'summary(regression.list)'

> more.complex.model[[1]]

eco.trait.regression regression
Method:  mantel
Randomisation:  mantel ; Permutations:  10
Observed slope:  0.12
Random slope mean +/-SD:  0.12  +/-  0
Observed model summary:
```

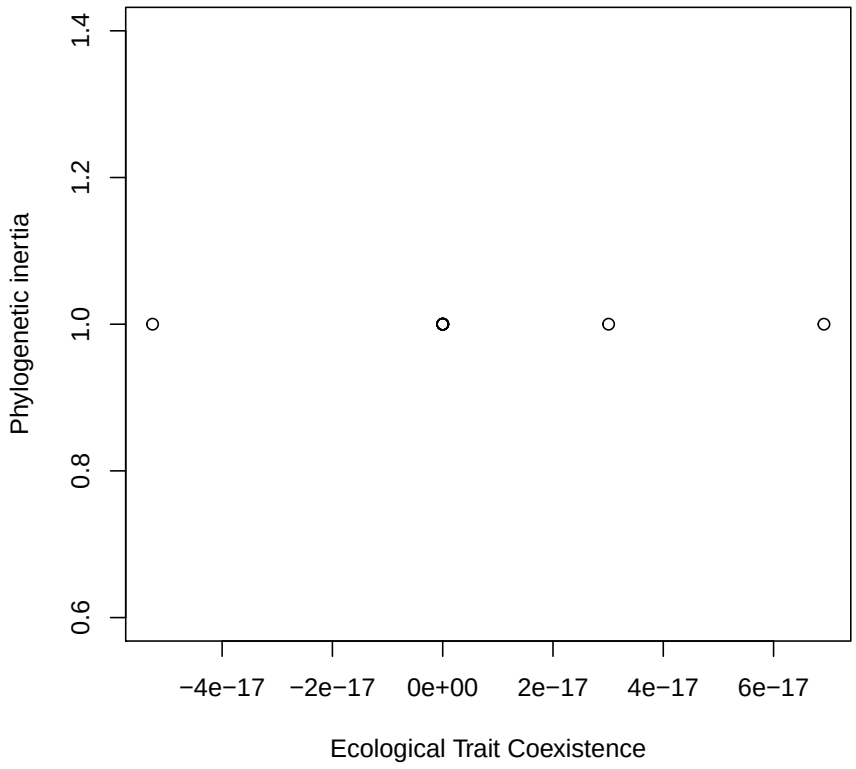
```
Mantel statistic based on Pearson's product-moment correlation

Call:
mantel(xdis = eco.mat, ydis = trait.mat)

Mantel statistic r: 0.1201
      Significance: 0.002

Upper quantiles of permutations (null model):
   90%    95%  97.5%    99%
0.0448 0.0580 0.0702 0.0802

Based on 999 permutations
```

```
> plot(eco.phy.regression(data, method="quantile"))
```



However, you can also regress the phylogenetic signal of traits in your dataset against the output from an `eco.trait.regression` on those traits. This allows you to generate figures similar to those of Cavendar-Bares et al. (2004; figure 4 XXX), only now with more modern measures of phylogenetic signal in traits (accessibly through the function `phy.signal`). For instance:

```
> model <- jcb.regression(data, eco.permute=100)
> plot(model)
```



6 Phylogenetic Generalised Linear Mixed Models

Matt, could you write this?