7.5.1 PGLS in R (caper)

Running PGLS in R (caper)

To begin, load the R package 'caper' (see Section 1.1.2 for installation instructions; Orme et al., in press):

```
library("caper")
```

Next download the data set and the phylogeny:

And then load the files into *R*:

```
primatedata <-read.table("Primatedata.txt", sep = "\t", header = TRUE)
```

```
primatetree <-read.nexus("consensusTree_10kTrees_Version2.nex")
```

Note that phylogenies in R cannot have spaces in the tip labels, instead R uses underscores to separate Genus and species (Genus_species). The names of the species in the tree must match those in the data, therefore in this dataset the spaces in species names have been replaced with underscores. This is easy to do in your own data:

```
primatedata$Binomial<-gsub(" ", "_", primatedata$Binomial)
```

caper requires your phylogeny and your data to be in a special kind of R object called a comparative data object. This will match the species names in the tree to those in your data automatically so there is no need to worry about data being ordered correctly etc. phy is your tree, data your data set, and names.col is the name of the column containing your species names.

```
primate <- comparative.data(phy = primatetree, data = primatedata, names.col = Binomial, vcv =
TRUE, na.omit = FALSE, warn.dropped = TRUE)
```

NOTE regarding the arguments to this function: vcv = TRUE stores a variance covariance matrix of your tree, which you will need for pgls. na.omit = FALSE stops the function from removing species without data for certain variables. warn.dropped = TRUE will tell you if there are any species which are not in the tree and the dataset and are therefore dropped from the data object.

If you drop species, you can see the list of dropped species by typing primate$dropped. Make sure you check this list is what you expected, it may reveal typos in your species names. If you want to turn this off use warn.dropped = FALSE. Note that the comparative.data function does all the matching of data and tree for you. Note that we expect to drop some species here because not all the species in primatetree are in primatedata and vice versa.

PGLS

The function for PGLS analyses in caper is pgls. To fit a model of log gestation length against log body mass which uses the maximum likelihood estimate of lambda we use the following code:

```
model.pgls<-pgls(log(GestationLen_d)~ log(AdultBodyMass_g), data = primate, lambda

='ML')
summary(model.pgls)
```

If you get an error related to optimization, you may need to adjust the bounds on the search of the maximum likelihood space. Each of the parameters that can be fit are given bounds, the default values for the bounds are lambda: 1e-6 to 1, kappa: 1e-6 to 3 and delta: 1e-6 to 3. To change the bounds, use the "bounds" argument within your pgls model, for example:

```
model.pgls<-pgls(log(GestationLen_d)~ log(AdultBodyMass_g), data = primate, lambda ="ML", bounds
= list(lambda=c(0.001,1), kappa=c(1e-6,3), delta=c(1e-6,3)))
```

Note that here we have changed the lower bound of lambda from 1e-6 to 0.001. At present if you want to change the bounds for any one parameter you have to list the bounds for each of the other parameters as well. Hence above we state the bounds for delta and kappa although we are not fitting these parameters with ML.

The output should look like this:

```
Call:
pgls(formula = log(GestationLen_d) ~ log(AdultBodyMass_g), data = primate,
    lambda = "ML")
Residuals:
      Min        1Q    Median        3Q       Max
-0.098899 -0.011661  0.003082  0.017758  0.075133
Branch length transformations:
kappa  [Fix]  : 1.000
lambda [ ML]  : 0.892
   lower bound : 0.000, p = 1.1435e-14
   upper bound : 1.000, p = 0.00046393
   95.0% CI   : (0.753, 0.967)
delta  [Fix]  : 1.000
Coefficients:
                    Estimate Std. Error t value  Pr(>|t|)
(Intercept)          4.290229   0.160355 26.7546 < 2.2e-16 ***
log(AdultBodyMass_g) 0.104864   0.019628  5.3426 9.479e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Residual standard error: 0.0261 on 75 degrees of freedom
Multiple R-squared: 0.2757,     Adjusted R-squared: 0.266
F-statistic: 28.54 on 2 and 75 DF,  p-value: 6.062e-10
```

As well as the standard regression outputs, the output includes the estimated ML value of (0.892) and p values from likelihood ratio tests showing whether the ML lambda is significantly different from 0 or 1 (see 5.3: Estimating Phylogenetic Signal - Pagel’s). In this case, the maximum likelihood estimate of is significantly different from both zero and 1; in other words, there is evidence for phylogenetic signal, but the trait has not evolved as expected under a Brownian motion model of evolution (given the branch lengths used).
We can also plot the results, including a plot of the likelihood surface of (which is well worth investigating - sometimes unexpected patterns are found, with peaks at 0 or 1!):

```
plot(log(GestationLen_d)~ log(AdultBodyMass_g), data = primatedata)

abline(model.pgls)
profile_lambda=pgls.profile(model.pgls, which="lambda") # vary lambda
plot(profile_lambda)
```

The parameters and are additional branch-length transformations implemented in caper that can improve the fit of the data to the tree (See 7.3: Common branch length transformations for more details). To optimise or , use the arguments delta = "ML" or kappa = "ML" instead of lambda = "ML" in the code above. Note that optimising more than one of these parameters at the same time is not advisable because it would be nearly impossible to interpret the results! Also note that for , you will need a different variance-covariance structure from that used above, specifically a 3D vcv.array. To do so, use the vcv.dim argument when constructing your comparative data object:

```
primate_3d <- comparative.data(phy = primatetree, data = primatedata, names.col = Binomial, vcv.
dim=3, vcv = TRUE, na.omit = FALSE, warn.dropped = TRUE)


model.pgls.kappa<-pgls(log(GestationLen_d)~ log(AdultBodyMass_g), data = primate_3d, kappa
="ML")
```

You should find that the maximum likelihood estimate of is 0.458, and that it also is significantly different from zero (as predicted under a speciational model of evolution) and one (as predicted under Brownian motion using the given branch lengths). Let's check out the likelihood surface:

```
profile_kappa=pgls.profile(model.pgls.kappa, which="kappa") # vary kappa
```

```
plot(profile_kappa)
```

Other considerations

Outliers

Outliers can seriously affect the parameter estimates for any regression model and PGLS models are no exception. Some researchers recommend the removal of outliers with studentized residuals >±3 (Jones and Purvis 1997). Note that caper does not at present have an automated outlier removal option (compare to crunch, section 7.2.2), but it may in the near future. Check ?pgls for updates.

To identify the outliers in a PGLS model, first you need to extract the phylogenetic residuals from the model:

```
res<- residuals(model.pgls, phylo = TRUE) #extracts phylogenetic residuals from the pgls model
```

Next these residuals need to be standardized by dividing by their square root of their variance:

```
res<- res/sqrt(var(res))[1] #standardises residuals by sqrt of their variance
```

Finally the residuals can be matched to the species names and the outlier idenitified (in this case Colobus_polykomos):

rownames(res)<-rownames(model.pgls$residuals) #matches the residuals up with the species names

```
rownames(res)[(abs(res)>3)]#gives the names of the outliers
```

You can then remove these species from the data and tree and redo the analysis. Note that you may need to continue removing species until there are no more outliers.

```
primate_nooutliers<-primate[-which(abs(res)>3),]

model.pgls_nooutliers<-pgls(log(GestationLen_d)~ log(AdultBodyMass_g), data = primate_nooutliers,
lambda='ML')

summary(model.pgls_nooutliers)

plot(model.pgls_nooutliers)
```

```
abline(model.pgls_nooutliers)
```

NOTE: These are phylogenetic residuals so they detect "phylogenetic outliers". Thus outliers may not obviously appear to be outliers on a plot of the raw data.

```

```

Model diagnostic plots

It is generally worth checking model diagnostic plots whenever you fit a model in R to check that your data meet the assumptions of linear modelling. The method for this is the same for OLS, independent contrasts and PGLS models (though the graphs are slightly different):

```
par(mfrow=c(2,2))#so you can see all 4 plots at once
plot(model.pgls)
```

```
par(mfrow=c(1,1))#to reset the graphic window to just one graph
```

Here are the model diagnostics for our model above:

Essentially what you are looking for in these plots are:

> *1) No studentised model residuals >±3. Any species with such large residuals should be removed. These outliers may overly influence the results of the regression.*
> *2) The points of the Q-Q plot should approximately form a straight line (rather than a banana shape).*
> *3 and 4) These should show a fairly random scattering of points. You want to avoid any clear patterns.*

It takes practice to know what is “good”, “bad” and “acceptable” with these plots. These plots look OK, but there appear to be several outliers; it may be worth investigating how these outliers affect the results.

References

> *Jones, K. E. & Purvis, A. 1997. An optimum body size for mammals? Comparative evidence from bats.* Funct. Ecol. ***11****: 751-756.*

> *Orme, C. D. L., Freckleton, R. P., Thomas, G. H., Petzoldt, T., Fritz, S. A. & Isaac, N. J. B. in press. caper: Comparative Analyses of Phylogenetics and Evolution in R.* Methods Ecol. Evol.

Contributed by Natalie Cooper and Charlie Nunn