GMM Tutorial

Katrina Jones & David Polly

2018-11-01

This tutorial includes code and excersizes used in the 2018 Analytical Paleobiology Course, hosted at University of Florida.
Code and datasets associated with this tutorial are available at
https://github.com/katrinajones/qpal
These can be installed with devtools using install_github("katrinajones/qpal", build_vignette=T)

Introduction to Geometric Morphometrics

Geometric morphometrics (GMM) is the quantitative representation and analysis of morphological shape using geometric coordinates instead of measurements.
The goal of morphometrics is to measure morphological similarity and difference. Key features of GMM are:

- Shape is represented by landmarks
- Variables are cartesian coordinates
- must be registered with procrustes analysis
- coordinate system has no objective scale - size not included
- results can be visualized as pictures
- there are several variants on the 'standard' analysis e.g., EDMA

Steps in a GMM analysis are:

- Collect landmark data
- Do a Procrustes superimposition
- Analyze similarity and difference of shape

Collecting landmarks

1. Each shape must have the same number of landmarks
2. The landmarks on all shapes must be in the same order
3. Landmarks are ordinarily placed on homologous points, points that can be replicated from object to object based on common morphology, common function, or common geometry.

Procrustes superimposition

Procrustes superimposition is the ‘standardization’ step in GMM. Procrustes removes size, translation, and rotation. In orther words, it centers the shapes on the same point, scales them to the same size, and rotates them into the same orientation. These manipulations remove statistical degrees of freedom, which has implications for later statistical analyses. After landmarks have been superimposed, the similarities and differences in their shape can be analyzed.
Other names for procrustes superimposition include: Procrustes analysis, procrustes fitting, generalized procrustes analysis (GPA), generalized least squares (GLS), least squares fitting. The analysis centers the shape on the origin, scales all shape to unite size, and minimizes least squares distance by rotation.

Principal components analysis (PCA)

PCA ordinates the objects by arranging them in a shape space. Similarities and differences can be easily seen in a PCA plot.
The axes of a PCA plot are principal components (PCs). The first PC is, by definition, the line that spans the largest axis of variation in shape. The second spans the next largest axis of variation at right angles to the first, the third spans the third largest axis of variation, and so on.
The PCA plot is a morphospace:

- each point represents a unique shape
- a shape model can be constructed for every point
- shape forms a continuous gradient through the PC space
- Each PC axis describes a different component of shape
- Real shapes are found where their landmarks correspond to the shape space
- Coordinates of the shapes are scores
- Each exis is orthogonal or uncorrelates
- Each axis accounts for a descending proportion of the total shape variance
- Positive/negative directions of PC axes are entirely arbitrary
- Axes are scaled in Procrustes units (meaningless except within-study)

Morphospace is always centered on the average shape, or consensus shape.
Why is PCA a standard step?

- Rotates data to its major axes for better visualization
- Preserves original distances between points, in other words it does not distort the variation in the data, but only if the covariance matrix is used (standard in GMM)
- Removed correlations between landmark coordinates and adjusts to proper degrees of freedom to simplify statistical analyses.

Visualization

Thin plate splines are a visualization tool which involves deforming a uniform grid to show a difference between two objects.
Lollipops show vector from each landmark in one shape to another, to visualize shape changes.

Definitions

Landmarks: ant point described by cartesian coordinates, used to represent the shape of a structure.
Landmarks (2): any point that can be places on a biologically or geometrically homologous point on the structure.
Semi-landmark: a point that is placed arbitrarily using an algorithm, often by defining endpoints at biologically homologous points and placing a specified number of semilandmarks between them.
Sliding semi-landmarks: semilandmarks points whose positions are algorithmically adjusted to minimize either the Procrustes distance or 'bending energy'. However, placement is sample dependent.
Surfaces: semilandmark representatikon of the 3D surface of an object. Semilandmarks are quantified as cartesian coordinates. Either ordinary (object-dependent) or sliding (sample dependent) semilandmarks.
Consensus shape: the mean of the procrustes coordinates, landmark by landmark
Centroid: the geometric center of the object (the average of all of the x/y/z)
Procrustes distance: the sum of all the distances betweent the corresponding landmarks of two shapes. The main measure of shape difference.
Centroid size: Sum of distances from landmarks to centroid, measure of overall size.

Advantages of GMM

1. Results can be visually represented as shapes
2. Data are easily collected from digital photographs
3. Size is mathematically removed from the analysis to focus on pure shape

Disadvantages of GMM

1. Size is absent from analysis, and size may be biologically relevant
2. Only single rigid structures can be easily analyzed
3. GMM cannot measure variation in a single landmark. Procrustes distributes variation by least-squares to minimize differences between whole shapes. Variation at an individual landmark cannot be interpreted as biological variation. Use EDMA instead.

How do you choose landmarks?

1. Data must reflect a hypothesis
2. Data must represent the shape adequately
3. Landmarks must be present on all specimens
4. Measurement error always exists in any collection of data, but ME doesn't matter if its substantially less than the differences you want to measure.
5. Sample size required for a particular study depends on the within-group variation relative

to differences between groups.

How many specimens do I need?

- Depends on the question!
- Depends on the error in the data
- You need more specimens when the differences you want to measure are small compared to the variation within your group

What morphometrics can NOT tell you

1. Morphometrics does not tell you what ‘large’ or ‘difference’ or ‘shape’ mean.
2. Morphometrics does not tell you wether you unwittingly have two unrecognizes groups in a single sample
3. How to identify cladistic characters.

Introduction to Geomorph Package

Geomorph provides a flexible tool for analyzing geometric morphometric data. For more information see ?(geomorph).
Loading up some data:

```
#load geomorph
library(geomorph, quietly=T)
library(qpal, quietly=T) #package including functions for the course

#Get example data - plethodon salamander heads
data("plethodon") #Load data into environment

#What's included in the dataset
summary(plethodon)
#>         Length Class  Mode
#> land     960   -none- numeric
#> links     28   -none- numeric
#> species   40   factor numeric
#> site      40   factor numeric
#> outline 7262   -none- numeric
```

The ‘wide’ data format (n*(p*k)) for data is most common outside GMM packages, however, within GMM packages the long format (p*k*n, 3D array) is most commonly used.
Converting between wide and long data formats:

```
#Converting between wide and long formats
#Long to wide
wide<-two.d.array(plethodon$land)
wide[1:5,1:5]
```

```
#>           [,1]     [,2]      [,3]     [,4]     [,5]
#> [1,]  8.893720 53.77644  9.268400 52.77072 5.561040
#> [2,]  8.679762 54.57819  8.935628 53.83027 5.451914
#> [3,]  9.805328 56.06903 10.137712 55.27961 6.647680
#> [4,]  9.637164 58.03294  9.952104 56.77318 6.109836
#> [5,] 11.035692 58.75009 11.335110 57.85184 8.255382
#Wide to long
long<-arrayspecs(wide, 12, 2)
long[,,1]
#>           [,1]     [,2]
#>  [1,]  8.89372 53.77644
#>  [2,]  9.26840 52.77072
#>  [3,]  5.56104 54.21028
#>  [4,]  1.87340 52.75100
#>  [5,]  1.28180 53.18484
#>  [6,]  1.24236 53.32288
#>  [7,]  0.84796 54.70328
#>  [8,]  3.35240 55.76816
#>  [9,]  6.29068 55.70900
#> [10,]  8.87400 55.25544
#> [11,] 10.74740 55.43292
#> [12,] 14.39560 52.75100
```

Arrays are much more handy for handling GMM data in R because you can easily access all elements with simple code

```
#Indexing arrays
#Get first specimen
long[,,1]
#Get first landmark of all specimens
long[1,,]
#Get X coordinates of all landmarks
long[,1,]
```

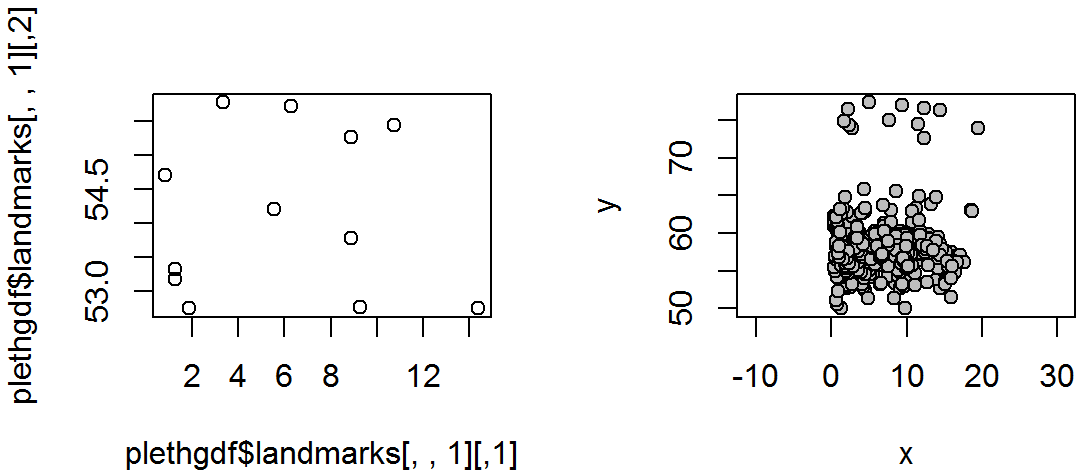
Geomorph data frames are a useful way of tidying all your data associated with your landmarks together into a single object, and are the native and sometimes required format for geomorph functions

```
#Create geomorph data frame
plethgdf<-geomorph.data.frame(landmarks=long,
species=plethodon$species,
                              site=plethodon$site)
summary(plethgdf)
#>           Length Class  Mode
#> landmarks 960    -none- numeric
#> species    40    factor numeric
#> site       40    factor numeric
```

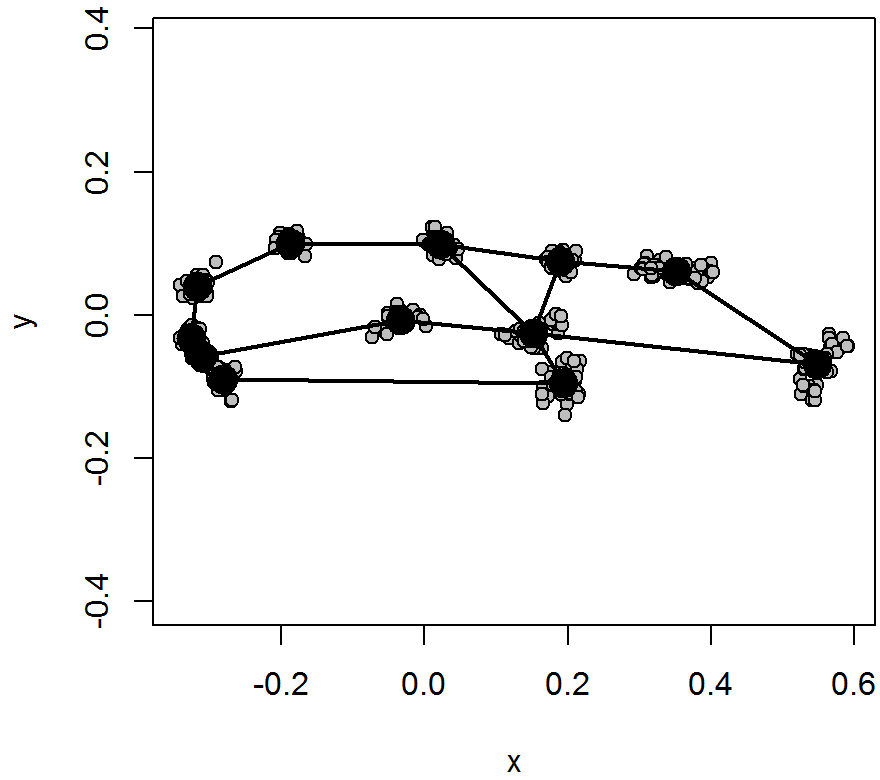
Once your data is all combined into a single geomorph data frame, its very useful to be able to subset it all at once, keeping track of all your separate variables. I've included a custom subsetting function I wrote for this purpose.

```
#plot first five specimens
plot(plethgdf$landmarks[,,1])
```

```
#plot raw data together
plotAllSpecimens(plethgdf$landmarks, mean=F)
```

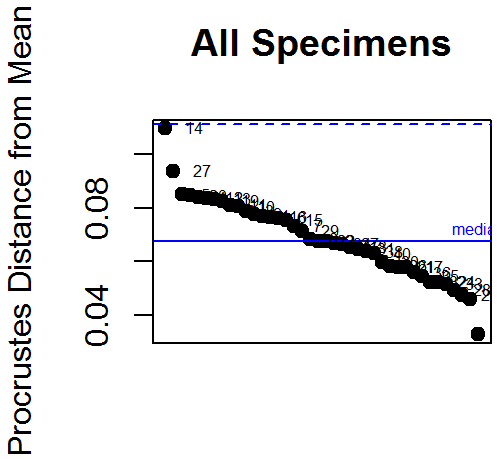



```
#plot in same shape space
Pcoords<-gpagen(plethgdf$landmarks)
plethgdf<-geomorph.data.frame(plethgdf, coords=Pcoords$coords,
                              size=Pcoords$Csize)
plotAllSpecimens(plethgdf$coords, mean=T, links=plethodon$links)
```



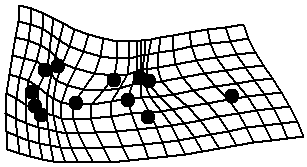
Comparing specimens

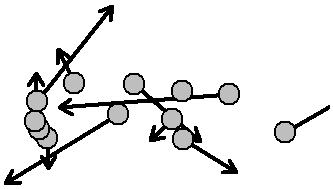
```
#Are there any wierd specimens
plotOutliers(plethgdf$coords)
```

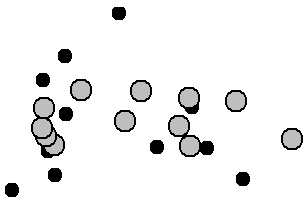



```
#Let's compare them
```

```
shape1<-mshape(plethgdf$coords)#consensus shape
shape2<-plethgdf$coords[,,14]

#Thin plate spline
plotRefToTarget(shape1, shape2, method=c("TPS"), mag=2)
#Lollipops
plotRefToTarget(shape1, shape2, method=c("vector"), mag=10)
#Points
plotRefToTarget(shape1, shape2, method=c("points"), mag=10)
```







Overview: Your first GMM analysis

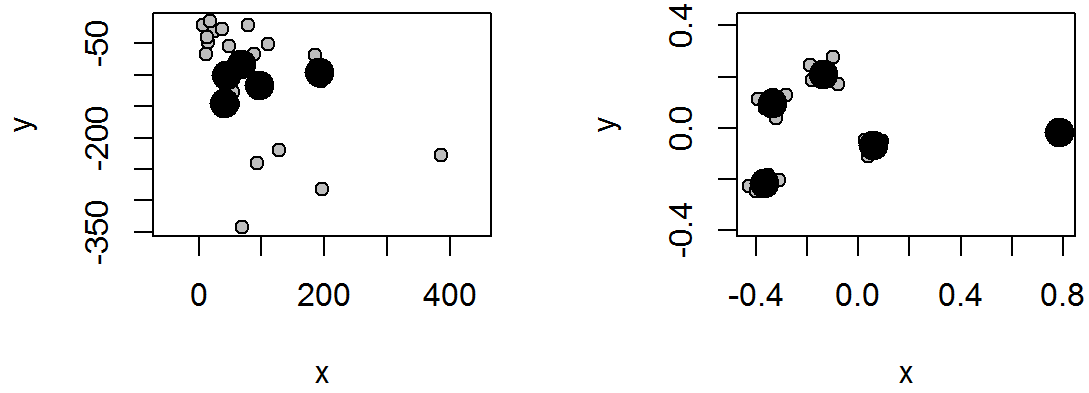
Step-by-step morphometric analysis

This section provides a quick overview of the steps of a GMM analysis in R, which will be followed by a practical in which you will collect and run your own data.
The R package Stereomorph provides a nice interface for collecting landmarks and curves in 2D. You must have either Chrome or Safari installed for this to work. For more guidence on collecting landmarks in Stereomorph see
https://aaronolsen.github.io/software/stereomorph/StereoMorph%202D%20Tutorial.pdf

```
library(StereoMorph)
#> Warning: package 'StereoMorph' was built under R version 3.4.4
library(abind)

#Digitize 2D landmarks with Stereomorph
extdatafiles <- system.file("extdata", package="qpal")#find directory
where external data are saved
setwd(extdatafiles)
#Name landmarks
lands<-c("condyle", "angular", "coronoid", "posterior molar", "tip
incisor")
#Name curves
curves<-matrix(c("condtocor",
"condtoang","condyle","condyle","coronoid",
                 "angular"), nrow=2,ncol=3)
#Digitize specimens
digitizeImage(image.file = 'ShrewsAndMarmots', shapes.file = 'teeth',
              landmarks.ref = lands, curves.ref = curves)
```

Read in the data

```
#Look at data based on fixed landmarks only
teeth<-StereoMorph::readShapes('teeth')
fixed<-teeth$landmarks.scaled
#Procrustes fit
gpa.lands <- gpagen(fixed)
plotAllSpecimens(fixed)
plot(gpa.lands)
```



Now add the curve sliders

```
 #With Sliding semi-landmarks
curves<-teeth$curves.scaled

#resample curves to 5 fixed landmarks
nfixed<-5
totfixed<-nfixed+2
curvelands<-list()
curvelands<-lapply(curves, function (x) lapply(x, pointsAtEvenSpacing,
n=totfixed))#Resample
curvelands<-lapply(curvelands, do.call, what=rbind)#merge
curvelands<-array(unlist(curvelands), dim=c((2*totfixed),2,5))#convert
to array

#The first and last points are the original fixed landmarks, so can be
dropped
curvelands<-curvelands[-c(1,totfixed,
(totfixed+1),(2*totfixed)),,]#remove fixed points
rownames(curvelands)<-rep(c("cu1", "cu2"), each=nfixed) #label
lands<-abind::abind(fixed, curvelands, along=1) #join with fixed
```

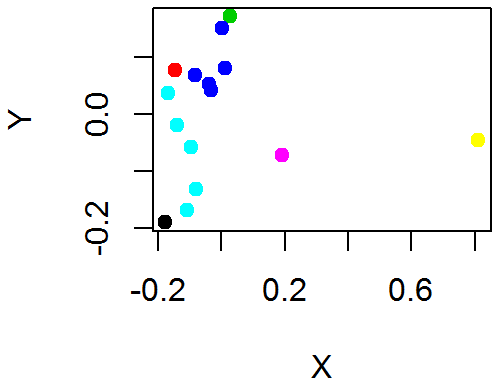
We can write to TPS format for use in other software, or read in files created elsewhere in that format

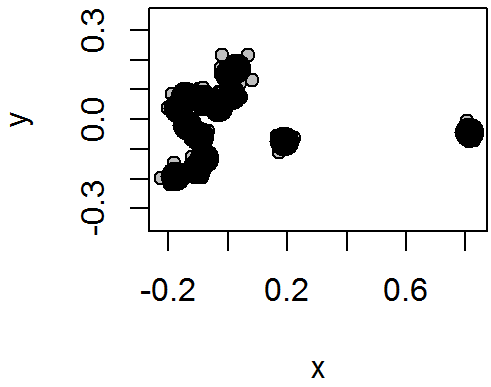
```
#Read and write TPS file for use in other software
writeland.tps(lands, "shrewtest.tps")
readland.tps("shrewtest.tps", specID=("ID"))
```

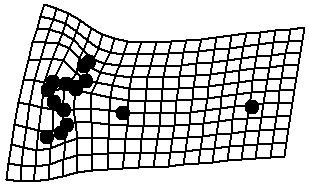
Now we need to make a ‘sliders’ file to tell geomorph which landmarks to slide during the procrustes fit. Here I’m using custom code makesliders which makes the sliders file based on the curve labels

```
#make sliders file
sliders<-makesliders(rownames(lands),id=c("cu1", "cu2"),
begin=c(1,1),end=c(3,2))
```

```
#Procrustes fit
gpa.lands <- gpagen(lands, curves=sliders)
plot(gpa.lands$consensus, col=palette()[as.factor(rownames(lands))],
pch=19)
plot(gpa.lands)
plotRefToTarget(gpa.lands$coords[,,1], gpa.lands$consensus)
```



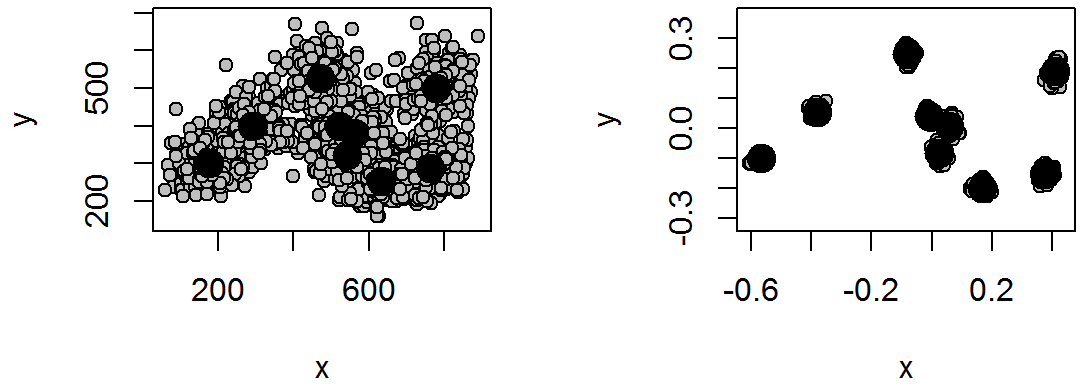




Now we'll try with a much bigger dataset, see ?shrewteeth

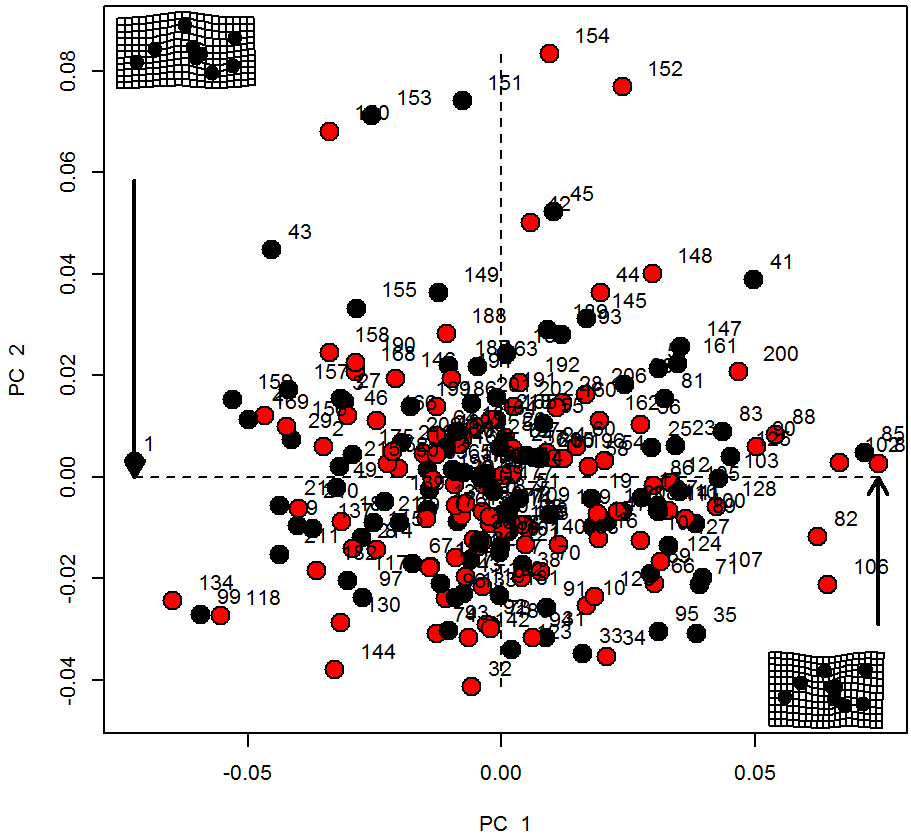
```
##Now lets try with a big dataset of shrewteeth!
#shrew teeth
data("shrewteeth")
summary(shrewteeth)
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>    60.0   314.0   442.0   450.1   555.0   891.0
#Procrustes fit
```

```
proc<-gpagen(shrewteeth)
#Plot raw data
plotAllSpecimens(shrewteeth)
#plot procrustes coordinates
plot(proc)
```



Now lets try putting it in a geomorph data frame

```
#Make geomorph data frame
group<-as.factor(rep(c("a", "b"), length=dim(shrewteeth)[[3]]))#make
up some labels
shrewdf<-geomorph.data.frame(raw=shrewteeth, coords=proc$coords,
                             size=proc$Csize, group=group)

#subsample by labels
a<-subsetgeom(shrewdf, "spec", which(shrewdf$group=="a"), keep=T)

#PCA
pca.lands <- plotTangentSpace(shrewdf$coords, label=TRUE,
                              groups = palette()[shrewdf$group])
```




```
shrewdf<-geomorph.data.frame(shrewdf, scores=pca.lands$pc.scores)
```

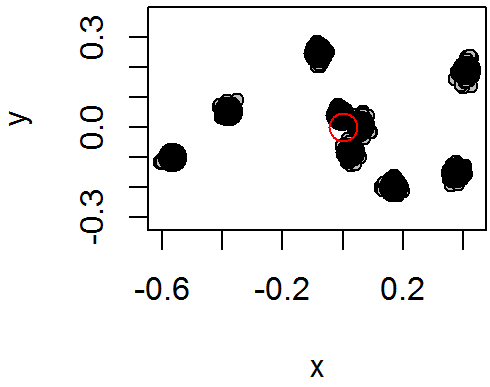
Now lets try an interactive 3D plot

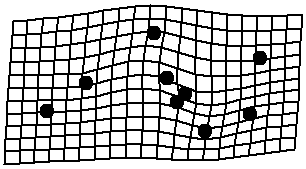
```
plot3d(shrewdf$scores[,1:3])#interactive 3D plots
text3d(shrewdf$scores[,1:3],texts=rownames(shrewdf$scores),pos=4,cex=.
5)
```

You must enable Javascript to view this page properly.

Compare to the mean shape

```
#Consensus shape
consensus <- mshape(shrewdf$coords)
plotAllSpecimens(shrewdf$coords)
#centroid - midpoint of the landmarks
centroid <- apply(shrewdf$coords,2,mean)
points(centroid[1],centroid[2],col="Red", cex=2)

#Visualizing shapes
plotRefToTarget(consensus,shrewdf$coords[,,1])
```





Practical - GMM analsysis of faces

Now try it out yourself! Take some photos of your friends faces, ideally from different angles, and try running your own morphometric analysis. Do different faces group separately? Or does error associated with photo angle swamp out the signal?

Getting into the weeds: GMM analysis in detail

Procrustes superimposition

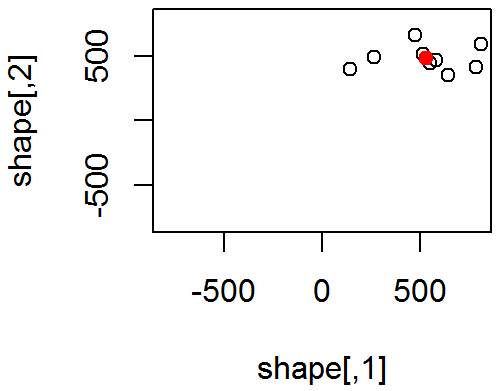
Procrustes superimposition translates, scales and rotates ladmarks to a common coordinate system. This removes degrees of freedom as follows:
2 + 1 + 1 = 4 for 2D data 3 + 1 + 1 = 7 for 3D data
This creates several statistical challenges: colinearity, ‘singular’ covariance matrix, and ‘curved’ shape space, which can be dealt with later in the PCA step.
Calculating Procrustes distances by hand

```
#Distances in shape space
#Calculate distance
A<-shrewdf$raw[,,1]
B<-shrewdf$raw[,,2]
pdist<-sqrt(sum((A-B)^2))
pdist
#> [1] 314.8666
```

```
#Consensus shape
consensus <- mshape(shrewdf$coords)

#Procrustes distance
dists <- array(dim=dim(shrewdf$coords)[3])
for(i in 1:dim(proc$coords)[3])
{ dists[i] <- sqrt(sum((shrewdf$coords[i]-consensus)^2)) }
head(sort(dists))
#> [1] 1.000005 1.000005 1.000220 1.000232 1.000314 1.000536
```

Calculate centroid size

```
#Centroid size
shape <- A
centroid <- apply(shape,2,mean)
centroidsize <- sqrt(sum((t(t(shape)-centroid))^2))

plot(shape, xlim=c(-800,800), ylim=c(-800,800))
points(t(centroid), col="red", pch=19)
```

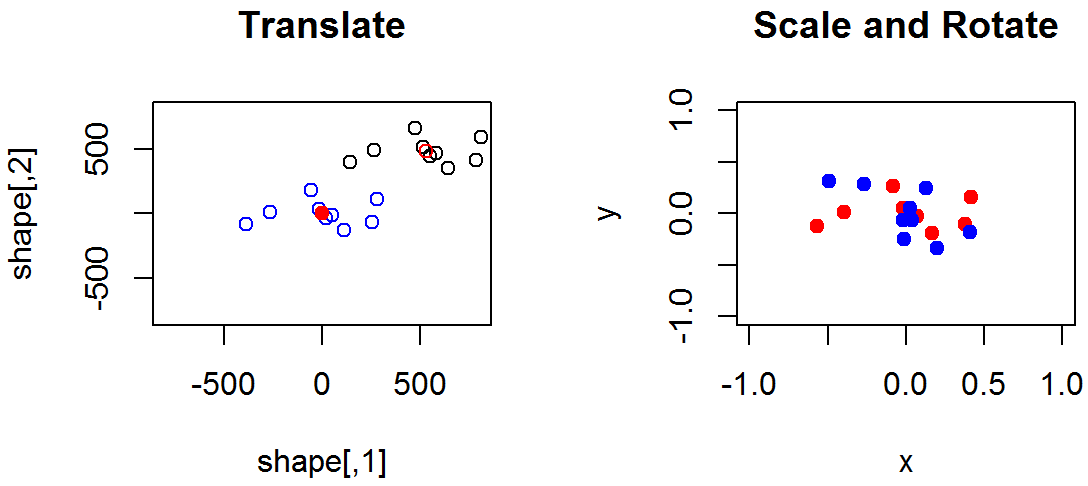


The following code allows you to run through the steps of a Procrustes Superimposition: Translation, scaling, rotation

Translation

```
#Translation
newshape<-t(t(shape)-centroid)

#plot
plot(shape, xlim=c(-800,800), ylim=c(-800,800), main="Translate")
points(newshape, col="blue")
points(t(centroid), col="red")
newcent<-apply(newshape, 2, mean)
points(t(newcent), pch=19,col="Red")

#centroid size doesn't change
```

```
newcentsize<-sqrt(sum((t(t(newshape)-newcent))^2))
centroid
#> [1] 531.6667 481.7778
newcentsize
#> [1] 678.6365

#scale
resizedshape <- newshape / centroidsize

#plot
plot(c(-1,1),c(-1,1),xlab="x",ylab="y", type="n", main="Scale and
Rotate")
points(t(newcent), pch=19)
points(resizedshape, pch=19,col="Red")

angle <- 45 * pi / 180
rotmat <-
matrix(c(cos(angle),-sin(angle),sin(angle),cos(angle)),byrow=T,ncol=2)

#plot
points(resizedshape%*%rotmat, pch=19,col="Blue")
```



Principal Components Analysis

One way to check your calculations is to compare three methods for calculating total variance which should always give the same results

```
nspec<-dim(shrewdf$raw)[3]
nspec
#> [1] 219
sum(apply(shrewdf$coords, 3, function (x)
```

```
sum((x-consensus)^2)))/(nspec-1)
#> [1] 0.002525195
sum(pca.lands$sdev^2)
#> [1] 0.002525195
sum(pca.lands$pc.scores^2)/(nspec-1)
#> [1] 0.002525195
```

The following code runs through the steps of the PCA calculation

```
#convert procrustes coordinates to wide format for use outside
geomorph
coords2d<-two.d.array(shrewdf$coords)
consensusvec<-apply(coords2d, 2, mean)#consensus in vector format

#calculate residuals
resids<-sweep(proc$coords, c(1,2), consensus)

#calculate covariance matrix
P<-cov(two.d.array(resids))

#single value decomposition
pca.stuff<-svd(P)
eigenvalues <- pca.stuff$d
eigenvectors <- pca.stuff$u
scores <- two.d.array(resids)%*%eigenvectors
```

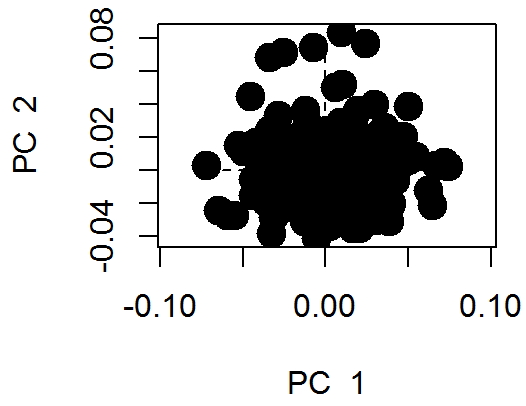
Calculate variances again to check

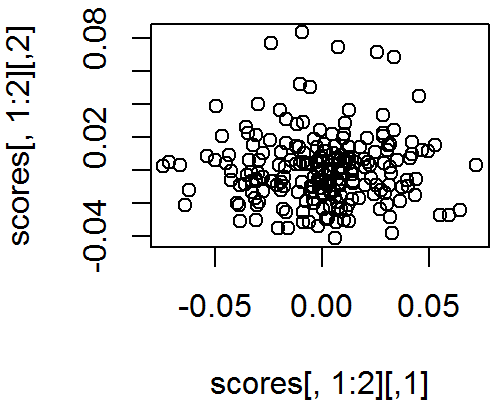
```
sum(apply(coords2d,2,var)) # total variance of Procrustes coordinates
#> [1] 0.002525195
sum(apply(two.d.array(resids),2,var)) # total variance of Procrustes
residuals
#> [1] 0.002525195
sum(pca.stuff$d) # total variance of singular values
#> [1] 0.002525195
sum(apply(scores, 2, var)) # total variance of scores
#> [1] 0.002525195
```

All looks good!

Compare plots to those generated by PCA using geomorph.

```
pca<-plotTangentSpace(shrewdf$coords, warpgrids=F)
plot(scores[,1:2])
```





Generating shapes from morphospaces

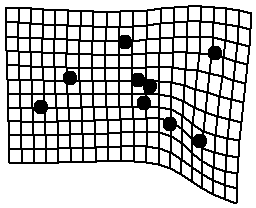
PCA is nothing more than a rigid rotation of the data, meaning we can translate between PCA scores and real shape. Eigenvectors are the angles between PC and original variables, making them in effect a rotation matrix. Therefore, we can translate between original data and PC scores by multiplying by the eigenvector rotation matrix.

```
###Translate PC Scores into original shapes
pcscores<-pca$pc.scores
eigenvectors<-pca$rotation
consensus<-mshape(shrewdf$coords)

#First lets calculate the consensus shape
scores<-c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)#PC scores of consensus shape
shape<-scores*eigenvectors
shape<-matrix(shape,nrow=9, ncol=2, byrow=T)#must do by row
#Shape object will be zero for middle of morphospace
shape
#>       [,1] [,2]
#>  [1,]    0    0
#>  [2,]    0    0
#>  [3,]    0    0
#>  [4,]    0    0
#>  [5,]    0    0
#>  [6,]    0    0
#>  [7,]    0    0
#>  [8,]    0    0
#>  [9,]    0    0

#Add consensus shape
```

```
shape<-shape+consensus
shape
#>               [,1]         [,2]
#>  [1,] -0.567840259 -0.099913381
#>  [2,] -0.379931044  0.053957251
#>  [3,] -0.082502039  0.247238604
#>  [4,] -0.008041951  0.040927010
#>  [5,]  0.024258117 -0.083400357
#>  [6,]  0.057227563  0.006661031
#>  [7,]  0.412126074  0.188724017
#>  [8,]  0.377682608 -0.154210724
#>  [9,]  0.167020932 -0.199983451
#> attr(,"class")
#> [1] "mshape" "matrix"
#compare
mshape(shrewdf$coords)
#>               [,1]         [,2]
#>  [1,] -0.567840259 -0.099913381
#>  [2,] -0.379931044  0.053957251
#>  [3,] -0.082502039  0.247238604
#>  [4,] -0.008041951  0.040927010
#>  [5,]  0.024258117 -0.083400357
#>  [6,]  0.057227563  0.006661031
#>  [7,]  0.412126074  0.188724017
#>  [8,]  0.377682608 -0.154210724
#>  [9,]  0.167020932 -0.199983451
#> attr(,"class")
#> [1] "mshape" "matrix"
```

We can use this same idea to calculate the shape at any position in morphospace, for example 0.5 on PC1 and 0.5 on PC2.

```
scores<-c(0.5,0.5,0,0,0,0,0,0,0,0,0,0,0,0)#PC scores of consensus
shape
shape<-scores*eigenvectors
shape<-matrix(shape,nrow=9, ncol=2, byrow=T)#must do by row

#Add consensus shape
shape<-shape+consensus
plotRefToTarget(consensus, shape)
```



Statistical analysis of GMM data

Ordination methods

PCA in GMM is (almost) always based on a covariance matrix. Correlation is similar to covariance but standardized by the spread (variance) of the data. Therefore, if a correlation matrix for PCA is used, the distribution of landmarks in space will be distorted.
PCA:

- shows variation in its natural scale
- used to show multivariate difference on small number of axes
- arranges data by major axes based on measured variables
- used in GMM to produce shape variables
- Axes are alinged along the major axes of variance
- distances in PCA space = procrustes distances

Canonical Variates Analysis (CVA), (aka Discriminant Function Analysis (DFA)):

- Used to identify differences between known groups
- discriminant functions are used to assign unknown objects to one of the groups
- can be combined with cross-validation experiments to produce stats on how well the varibales are able to classify objects
- axes are aligned to group means
- distances in CVA space = Mahalonobis distances (standardsized distance)
- having too many variables (PCs) may cause CVA to overestimate differences between groups, so may be best to only include a few PCs.

Principal Coordinates Analysis (PCO) (==MDS):

- arranges data by major axes based on distance measures
- not normally used with GMM because quantitative variables are always available

- PCO on euclidean distances = PCA on variables

Non-metric Multidimensional Scaling (NMDS):

- Arranges data so the distances on 2d plot are as similar as possible to original multivariate distances
- Does NOT preserve between-oject distances
- for visualization only, DO NOT analyze.
- PCO is sometimes referred to as multidimensional scaling, even though it is a totally separate approach

Between-groups PCA (BG-PCA):

- Projection of all data into a PCA based on group means
- Increasingly popular, but dubious
- In GMM can produce highly misleading plots in which overlapping groups appear distinct
- BG-PCA does not have the properties of ordinary PCA (most of the variance may be on the higher axes, axes will be correlated)
- CVA is preferred if the goal to identify group differences

Regression analysis

Regression measures the relationship between geometric shape and another continuous variables.

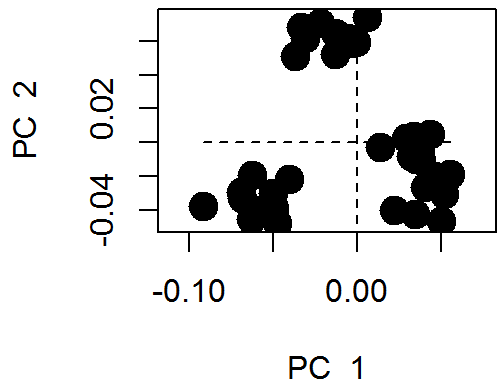
Linear regression finds the line that best predicts Y (dependent) from X (independent), where a is the slope, b is the intercept and E is the residual error.

$y=aX + b + E$

In GMM, we will always be using multivariate regression because there are always multiple shape variables.

We will use the plethodon example dataset from above to try out some data analysis in geomorph, using the geomorph dataframe 'plethgdf'.

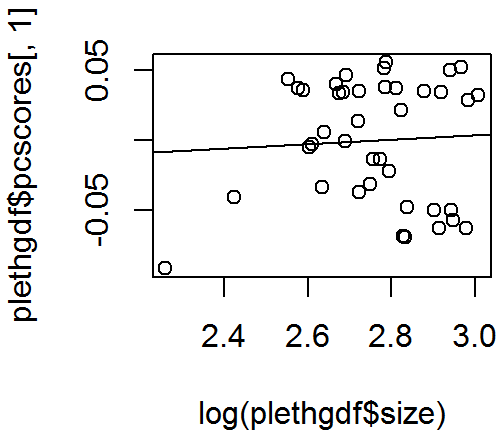
```
##example with plethodon dataset
pc.scores<-plotTangentSpace(plethgdf$coords, warpgrids=F)
```




```
plethgdf<-geomorph.data.frame(plethgdf, pcscores=pc.scores$pc.scores)
```

We can run the analysis univariately using PC1, however this is not a good idea because it ignores much of the variation. Here we show an example, just to illustrate univariate regression in r

```
#Univariate regression against PC1
#Not including all the variation - shouldn't do this!
plot(log(plethgdf$size),plethgdf$pcscores[,1])
w <- lm(plethgdf$pcscores[,1]~ log(plethgdf$size))
w # gives only coefficients & formula
#>
#> Call:
#> lm(formula = plethgdf$pcscores[, 1] ~ log(plethgdf$size))
#>
#> Coefficients:
#>         (Intercept)  log(plethgdf$size)
#>            -0.04360             0.01579
summary(w) # gives much more info: stats, P, etc.
#>
#> Call:
#> lm(formula = plethgdf$pcscores[, 1] ~ log(plethgdf$size))
#>
#> Residuals:
#>      Min       1Q   Median       3Q      Max
#> -0.08342 -0.03543  0.01080  0.03647  0.05491
#>
#> Coefficients:
#>                     Estimate Std. Error t value Pr(>|t|)
#> (Intercept)        -0.04360    0.12061  -0.361    0.720
#> log(plethgdf$size)  0.01579    0.04362   0.362    0.719
#>
#> Residual standard error: 0.04356 on 38 degrees of freedom
#> Multiple R-squared:  0.003438,   Adjusted R-squared:  -0.02279
#> F-statistic: 0.1311 on 1 and 38 DF,  p-value: 0.7193
abline(w) # adds regression line to plot
```



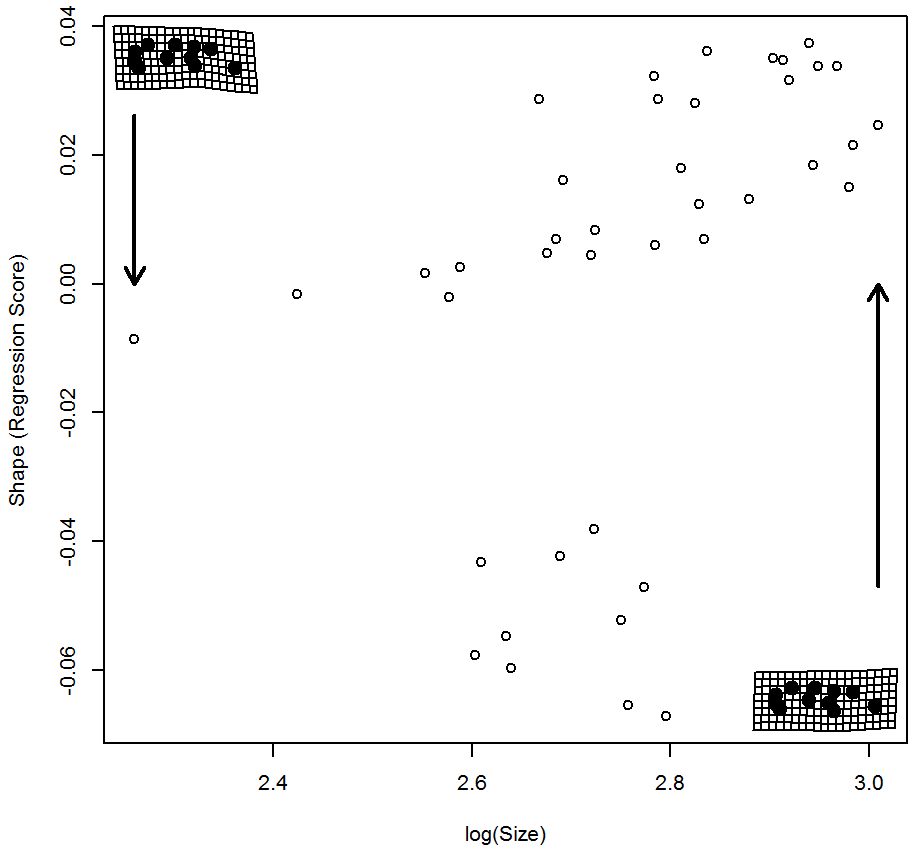
However, it is much better to run the analysis multivariately, taking into account all variation. Geomorph has many tools for the multivariate analysis of shape, which run directly on the Procrustes Coordinates. Further, statistical tests within the package are designed to accomodate high dimensional data. For more information see ?ProcD.lm

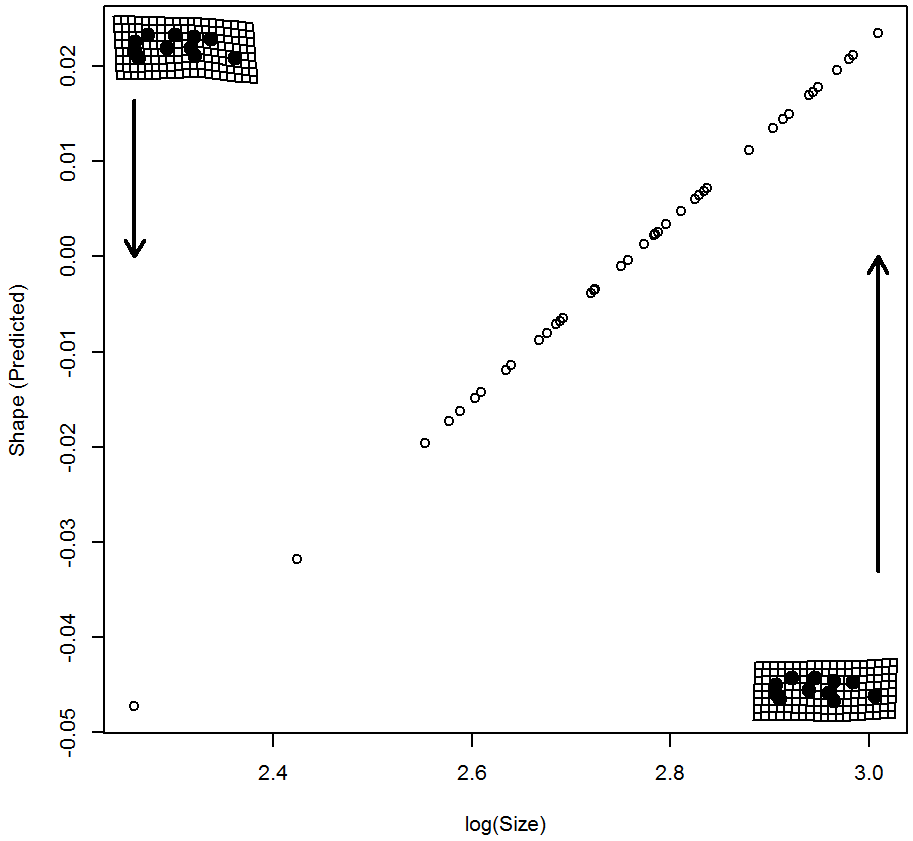
```
#Multivariate in geomorph - correct way!
pleth.lm<-procD.lm(coords~log(size), data=plethgdf, print.progress =
F)
pleth.lm
#>
#> Call:
#> procD.lm(f1 = coords ~ log(size), data = plethgdf, print.progress =
F)
#>
#> Type I (Sequential) Sums of Squares and Cross-products
#> Randomized Residual Permutation Procedure Used
#> 1000 Permutations
#> ANOVA effect sizes and P-values based on empirical F distributions
#>
#>
#>            Df       SS        MS     Rsq      F      Z Pr(>F)
#> log(size)   1 0.008894 0.0088940 0.04516 1.7973 1.2036  0.123
#> Residuals 38 0.188046 0.0049486 0.95484
#> Total     39 0.196940
```

If you are interested in the relationship between size and shape, there is also a built in function for allometric analysis. Though testing is essential the same as ProcD.lm, it includes additional tools for visualizing size-related shape.

```
#Special function for examining allometry specifically
pleth.allom<-procD.allometry(coords~size,logsz=T,data=plethgdf,
print.progress = F)
#>
#> Allometry Model
```

```
pleth.allom
#>
#> Call:
#> procD.allometry(f1 = coords ~ size, logsz = T, print.progress = F,
#>     data = plethgdf)
#>
#>
#>
#> Type I (Sequential) Sums of Squares and Cross-products
#> Randomized Residual Permutation Procedure Used
#> 1000 Permutations
#> ANOVA effect sizes and P-values based on empirical F distributions
#>
#>
#>           Df       SS        MS     Rsq      F      Z Pr(>F)
#> log(size)  1 0.008894 0.0088940 0.04516 1.7973 1.2036  0.123
#> Residuals 38 0.188046 0.0049486 0.95484
#> Total     39 0.196940

#Now also outputs allometric components with shape warps
#Shape scores from regression of shape on size
plot(pleth.allom, method="RegScore")

#Predicted shapes based on size
#PC1 of predicted values
plot(pleth.allom, method="PredLine")
```


Shape (Regression Score)
0.04
0.02
0.00
-0.02
-0.04
-0.06
2.4
2.6
2.8
3.0
log(Size)



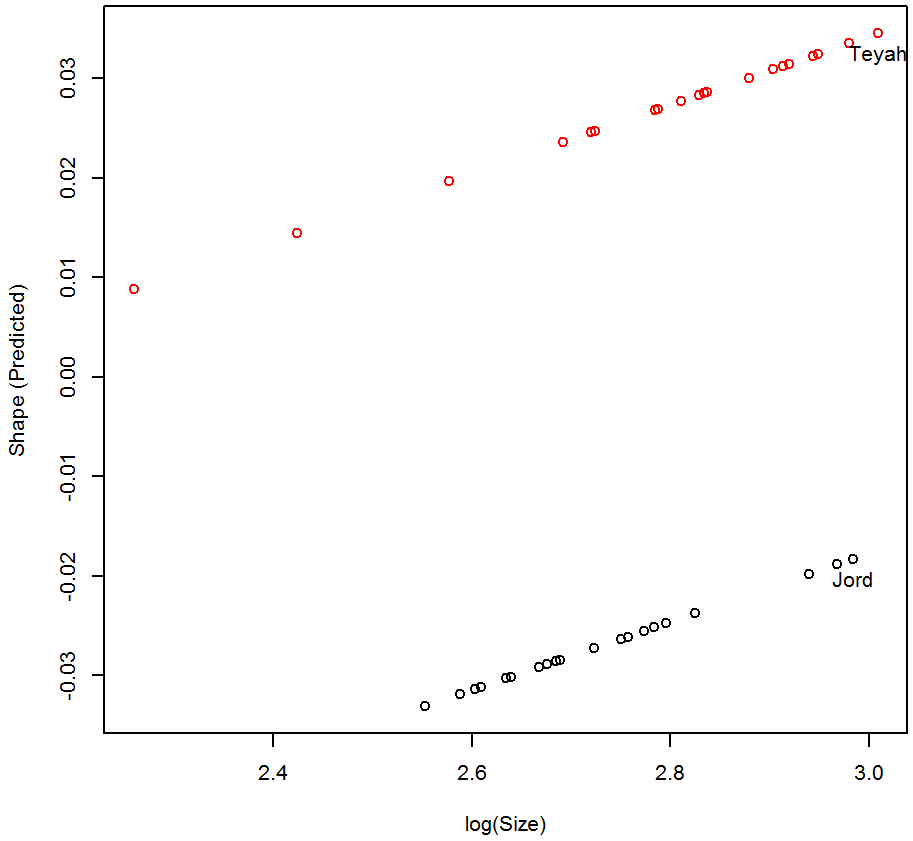
Further, it is possible to 'size-correct' your shape data, by removing the shape most associated with size. Remember, absolute size has already been removed during the Procrustes fit. Thus, this allometric component of shape reflects the remaining shape which is correlated with size. You need to think about if you want to remove this or not!

```
#Can get size-corrected residuals
plethAnova <- procD.lm(pleth.allom$formula,
                       data = pleth.allom$data, print.progress = F)
shape.resid <- arrayspecs(plethAnova$residuals,
                          p=dim(plethgdf$coords)[1],
k=dim(plethgdf$coords)[2]) # allometry-adjusted residuals
adj.shape <- shape.resid + array(mshape(plethgdf$coords),
dim(shape.resid)) #
```

We can also take into account different groups when considering allometry

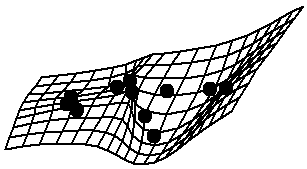
```
##Add groups - MANCOVA
pleth.allom<-procD.allometry(coords~size,f2=~species,logsz=T,data=plet
hgdf, print.progress = F)
#>
#> Homogeneity of Slopes Test
#>
#> Allometry Model
```

```
plot(pleth.allom, method="PredLine")
```



We can also visualize shape variation associated with size

```
#Compare min and max size associated shapes
plotRefToTarget(pleth.allom$Ahat.at.min, pleth.allom$Ahat.at.max,
mag=5)
```



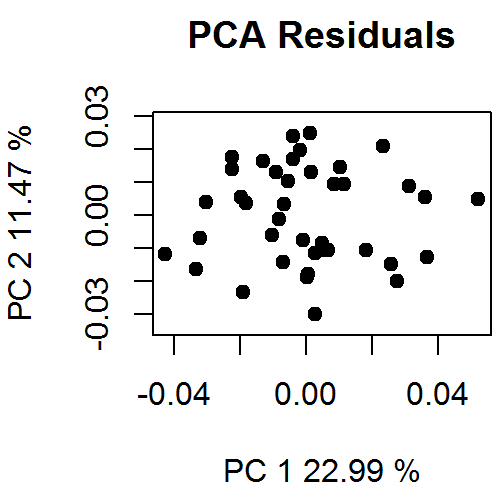
MANOVA

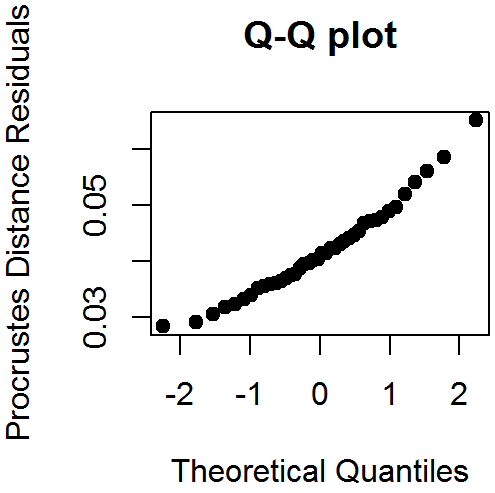
Assesses the relationship between geometric shape and a categorical predictor variable, with multiple response variables (shape).
We can also use ProcD.lm to examine the relationship of shape with categorical variables

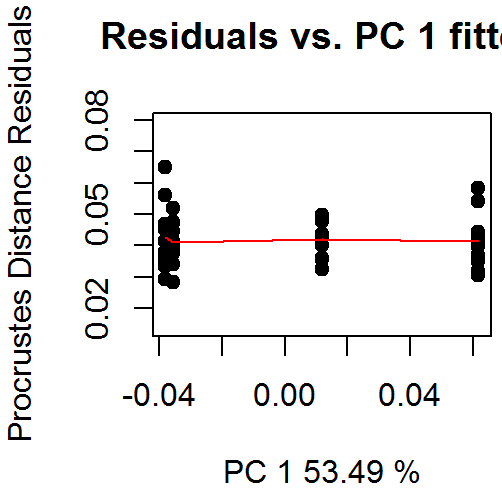
```
#ANOVA
pleth.anova<-procD.lm(coords~species, data=plethgdf, print.progress =
F)
pleth.anova$aov.table
#>           Df       SS        MS     Rsq      F     Z Pr(>F)
#> species    1 0.029258 0.0292578 0.14856 6.6304 3.296  0.001 **
#> Residuals 38 0.167682 0.0044127 0.85144
#> Total     39 0.196940
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
pleth.anova<-procD.lm(coords~species+site,
data=plethgdf,print.progress = F)
pleth.anova$aov.table
#>           Df       SS       MS     Rsq      F      Z Pr(>F)
#> species    1 0.029258 0.029258 0.14856 10.479 3.9884  0.001 **
#> site       1 0.064375 0.064375 0.32688 23.056 5.4594  0.001 **
#> Residuals 37 0.103307 0.002792 0.52456
#> Total     39 0.196940
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
pleth.anova<-procD.lm(coords~species*site, data=plethgdf,
print.progress = F)
pleth.anova$aov.table
#>              Df       SS       MS     Rsq      F      Z Pr(>F)
```

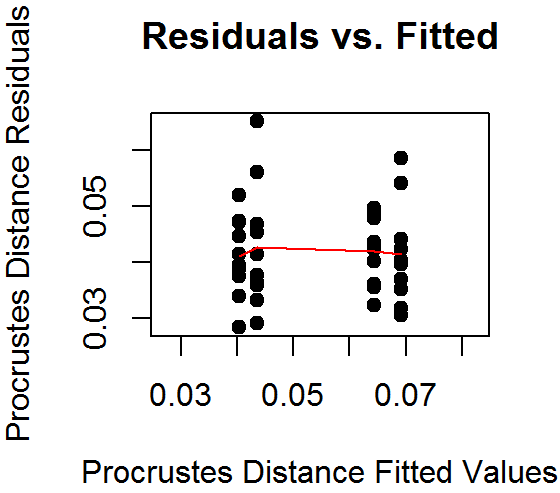
```
#> species       1 0.029258 0.029258 0.14856 14.544 4.4828  0.001 **
#> site          1 0.064375 0.064375 0.32688 32.000 5.9466  0.001 **
#> species:site  1 0.030885 0.030885 0.15682 15.352 6.8745  0.001 **
#> Residuals    36 0.072422 0.002012 0.36774
#> Total        39 0.196940
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

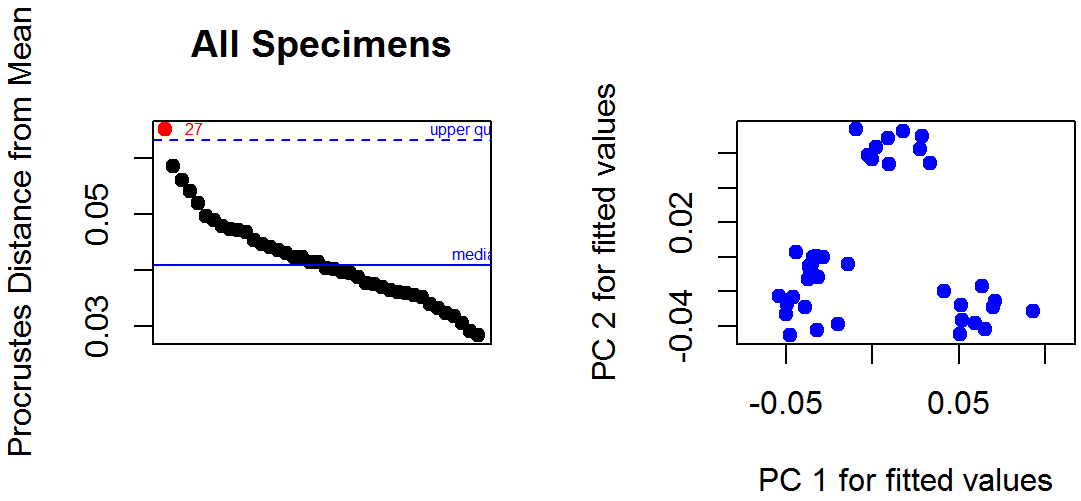
```
# diagnostic plots, including plotOutliers
plot(pleth.anova, type = "diagnostics", outliers = TRUE)
# PC plot rotated to major axis of fitted values
plot(pleth.anova, type = "PC", pch = 19, col = "blue")
```











To run posthoc tests between groups, or explicitly test the fit of two nested models, advanced.ProcD.lm can be used.

```
#Post-hoc comparisons
pleth.posthoc<-advanced.procD.lm(coords~species,f2=~1,
groups=~species,
                                 data=plethgdf, print.progress = F)
pleth.posthoc
#>
#> Call:
#> advanced.procD.lm(f1 = coords ~ species, f2 = ~1, groups =
~species,
#>     print.progress = F, data = plethgdf)
#>
#>
#>
#> Randomized Residual Permutation Procedure Used
#> 1000 Permutations
#> ANOVA effect sizes and P-values based on empirical F distributions
#>
#> ANOVA Table
#>
#>           ResDf     RSS       SS       MS     Rsq      F     Z
P
#> ~1 (Null)    39 0.19694
#> ~species     38 0.16768 0.029258 0.029258 0.14856 6.6304 3.296
0.001
#> Total        39 0.19694
#>           Pr(>F)
#> ~1 (Null)
#> ~species
#> Total         39
```

```
#>
#>
#> LS means
#>            [,1]         [,2]       [,3]         [,4]         [,5]
[,6]
#> Jord  0.1597260 -0.01869407 0.1929248 -0.08868376 -0.03102401
-0.002242806
#> Teyah 0.1449387 -0.03177919 0.1936548 -0.10139889 -0.03637704
-0.011616966
#>             [,7]         [,8]        [,9]        [,10]       [,11]
[,12]
#> Jord  -0.2800987 -0.09437595 -0.3109811 -0.06244151 -0.3257754
-0.03732346
#> Teyah -0.2835498 -0.08436582 -0.3104722 -0.05322464 -0.3262250
-0.02684087
#>            [,13]       [,14]       [,15]       [,16]       [,17]
[,18]
#> Jord  -0.3215131 0.03791574 -0.1881679 0.10172892 0.01989675
0.10464379
#> Teyah -0.3136324 0.04219763 -0.1883206 0.09896532 0.02328874
0.09306291
#>           [,19]       [,20]     [,21]       [,22]     [,23]
[,24]
#> Jord  0.1891566 0.08048506 0.357112 0.06419675 0.5387440
-0.08520871
#> Teyah 0.1897792 0.06939520 0.347150 0.05883370 0.5597657
-0.05322839
#>
#> LS means distance matrix
#>              Jord      Teyah
#> Jord  0.00000000 0.05409051
#> Teyah 0.05409051 0.00000000
#>
#> Effect sizes (Z)
#>           Jord    Teyah
#> Jord  0.000000 4.901917
#> Teyah 4.901917 0.000000
#>
#> P-values
#>        Jord Teyah
#> Jord  1.000 0.001
#> Teyah 0.001 1.000

#group and covariate
pleth.posthoc<-advanced.procD.lm(coords~species*log(size),f2=~1,
groups=~species*site, slopes= ~log(size), data=plethgdf,
print.progress = F)
pleth.posthoc$anova.table
```

```
#>                       ResDf     RSS      SS       MS     Rsq      F
Z
#> ~1 (Null)               39 0.19694
#> ~species * log(size)    36 0.15462 0.04232 0.014106 0.21489 3.2844
3.1924
#> Total                   39 0.19694
#>                          P Pr(>F)
#> ~1 (Null)
#> ~species * log(size) 0.001
#> Total                          39

#Compare with or without interaction term
pleth.posthoc<-advanced.procD.lm(coords~species*site,f2=~species+site,
groups=~species*site, data=plethgdf, print.progress = F)
pleth.posthoc$anova.table
#>                         ResDf      RSS       SS       MS     Rsq
F
#> ~species + site (Null)    37 0.103307
#> ~species * site           36 0.072422 0.030885 0.030885 0.15682
15.352
#> Total                     39 0.196940
#>                             Z     P Pr(>F)
#> ~species + site (Null)
#> ~species * site        6.8745 0.001
#> Total                                 39
```

Practical - Analysis of Carnivoran Tarsals (2D)

Now try for yourself, using an example dataset of photos of Carnivoran tarsals, in the folder 2Dtarsals. Locate this directory using system.file("extdata", package="qpal"). Associated data are located in Tarsalsdata.csv. To analyze example landmarks, there is a ready-made geomorph dataframe which can be accessed using data("tarsalsdf"). What is the influence of size and stance on tarsal shape in carnivores?

Visualizing 3D data in R

Geomorph offers great tools for visualizing shape change in 3D. Using a surface ply file (located in /Tarsals/plys), we can warp the surface shape to our procrustes consensus shape. From there, we can use this surface to visualize shape variation
First choose a specimen to be the base of your warps. This specimen should have fairly generic morphology and lie near the middle of the morphospace.

```
#3D surface warps
#Choose specimen
specfit<-c("Lynx_rufus")
```

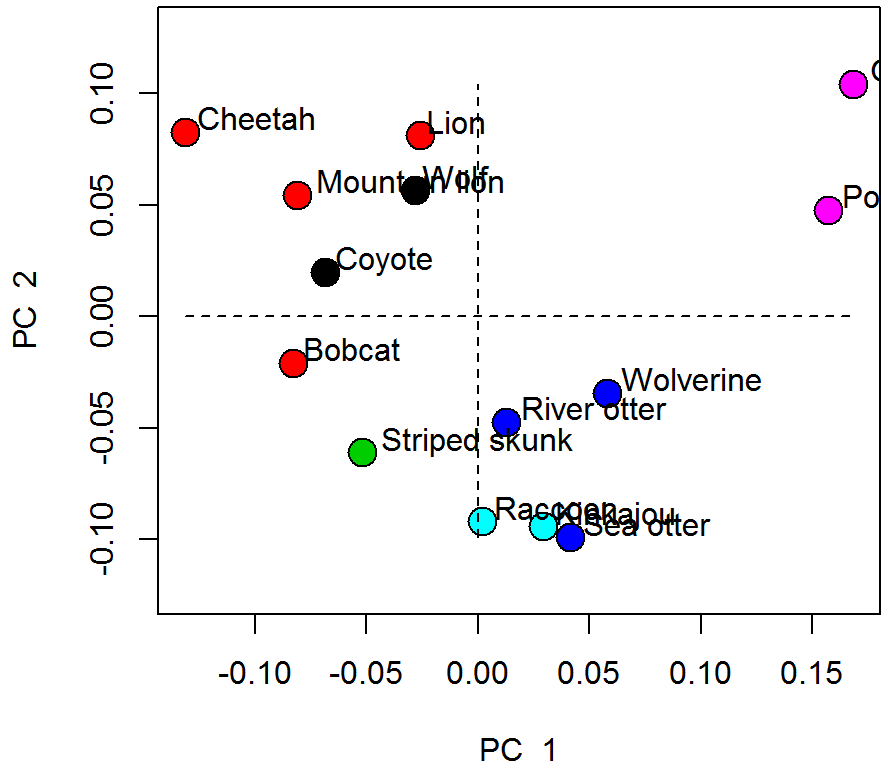
Next we will warp this base specimen to the mean shape for the dataset

```
#calculate warps
mean<-mshape(tarsalsdf$coords3d)

#read surface ply file
surf<-read.ply(paste0("Tarsals/plys/", specfit, ".ply"))

#warp
wmesh<-warpRefMesh(surf,tarsalsdf$land3d[,,which(dimnames(tarsalsdf$co
ords3d)[[3]]==specfit)], ref=mean)
```

You must enable Javascript to view this page properly.

Now we can use the wmesh object anywhere geomorph calls for a mesh. This will allow us to visualize shape variation as surface warps

```
#visualize shape
plotTangentSpace(tarsalsdf$coords3d, mesh=wmesh, label =
tarsalsdf$common, groups = tarsalsdf$family)
```



You must enable Javascript to view this page properly.

Analysis of 3D Tarsals

How can our estimates of shape change if we use 3D instead of 2D data? Access an example set of 3D landmarks using tarsalsdf$lands3d.

GMM on Trees

Phylogenetic comparative methods

Modified statistical tests that take into account non-independence in data that come from different species.
PCMs are applied to regression, MANOVA, and other standard tests, and they can be used to estimate rates of evolution and reconstruct ancestral traits on a phylogeny.
PCMs estimate how much covariance a trait should have between taxa (and traits) is expected from the phylogenetic topology. The expected phylogenetic covariance is removed, leaving the residual between-trait covariance. Statistical tests are carried out on the residual component.
PCMs based on a null model of evolution: Brownian motion i.e., purely random evolution.
Brownian motion is equivalent to an unbiased random walk. Change at each step is random with respect to other steps. Change at each step has an equal chance of moving in a positive or negative direction. Typical implementation specifies steps as coming from a normal distribution with mean of zero and variance equal to the rate of evolution.
In BM:

- the probability of endpoints is a normal distribution
- the most likely endpoint is the starting point
- the standard deviation increases with the square root of time
- the variance of the endpoints increases linearly with time
- the variance of the endpoints equals average squared change per step (rate) times number of steps (time)
- Because variance increases linearly with time, covariance between tips is linear with respect to shared branch time

Phylogenetic covariance matrix (C) has diagonal equal to the total branch length between tip and base and off diagonals equal to the length of the shared branches. Actual expected variance and covariance is C multiplied by rate.
Brownian motion makes a good statistical null because it is a purely random process. Outcomes of BM processes are statistically predictable. BM can occur in nature through genetic drift or selective drift (selection that changes randomly in direction and magnitude); therefore BM does NOT necessarily equate with ‘neutral evolution’.

Measuring phylogenetic signal

The phylogenetic component of data can only be assessed relative to some expectation e.g.,

BM.
Blombergs K (Kappa) = observed cov/expected cov
K can be thought of as the proportion of the covariance that is due to phylogeny. Usually ranges from 0 to 1, but can be greater than 1 if the phylogenetic covariances are stronger than under BM.
Pagels lambda = scaling factor so that tree fits BM model
Lambda can be thought of as how you would have to scale the branch lengths so that the data would be obtained under a BM model. Also usually ranges 0 to 1, where 0 is the equivalent to no phylogenetic structure and 1 is equal to actual phylogenetic structure. Can also be greater than 1 if phylogenetic covariances are stronger than under BM.

Ancestral state reconstruction

If the likelihood of ancestor of one descendant is a normal distribution with variance proportional to time, then the likelihood of the common ancestor is the product of their probabilities. This is the maximum likleihood method for estimating phylogeny, and reconstructing ancestral phenotypes. This can be used to project a phylogenetic tree into a morphospace (phylomorphospace).

PCMs in GMM

1. Phylogenetic least squares:
- Calculates C matrix to correct covriance structure in a least-squares regression
- Analogous to regressing out phylogenetic covariance as part of a trait-on-trait regression
- matches PIC under BM
2. Phylogenetic independent contrasts
- Flexible method in which data are transformed to change along non-overlapping segments of a tree
- analogous to ‘first differencing’ in time series
- contrasts are calculated for each trait of interest, then treated as phylogeny-free variables in ordinary statistical tests
3. Phylogenetic MANOVA
- Essentially a MANOVA on independent contrasts for testing
4. Phylogenetic MANCOVA
- Extension of PGLS for assessing group differences holding a continuous covariate constants

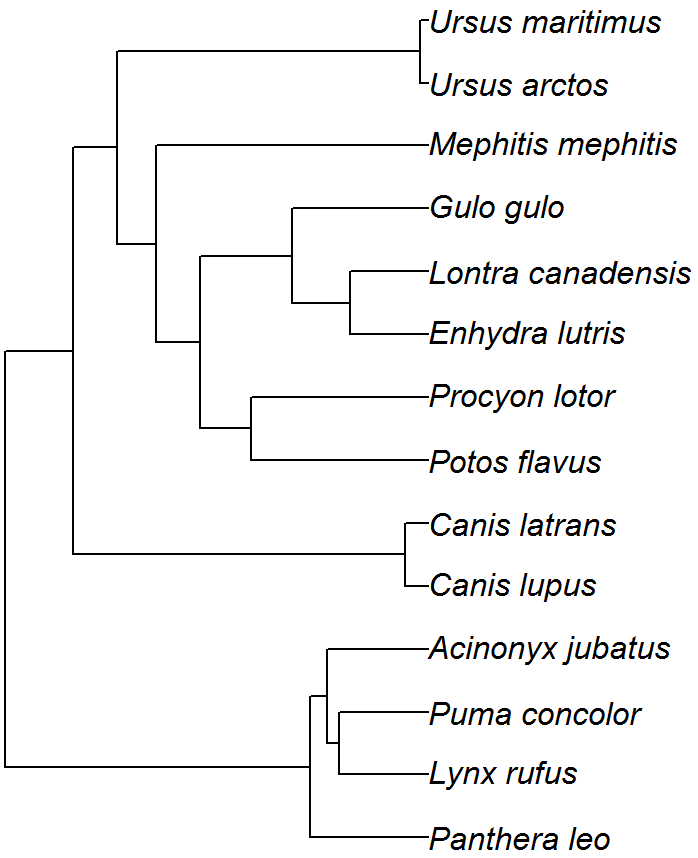
Phylogenetic PCA: - pPCA is a PCA-like analysis based on the adjusted covariance matrix (by C matrix) - has undesirable properties: does NOT remove the effects of phylogeny - recommend not using this for GMM

Applying in R

First we will upload a phylogenetic tree to match our tarsals dataset. This is a time callibrated

phylogeny generated by timetree.org.

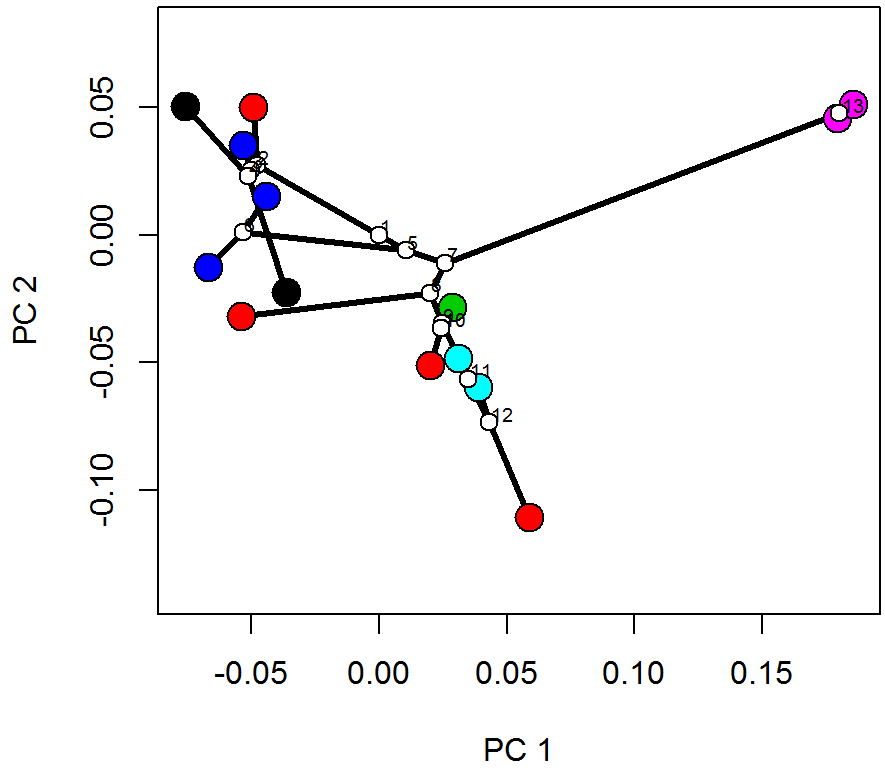
```
#Use 2D tarsal data from yesterday
data("tarsalsdf")

#Read tree - created from timetree.org
require(phytools)
#> Loading required package: phytools
#> Warning: package 'phytools' was built under R version 3.4.4
#> Loading required package: maps
#> Warning: package 'maps' was built under R version 3.4.4
tree<-read.newick("tarsaltree.nwk")
tree$tip.label<-gsub("_", " ", tree$tip.label)#make names match
par(mar=c(1,1,1,1))
plot(tree)
```



Visualizing evolution on trees

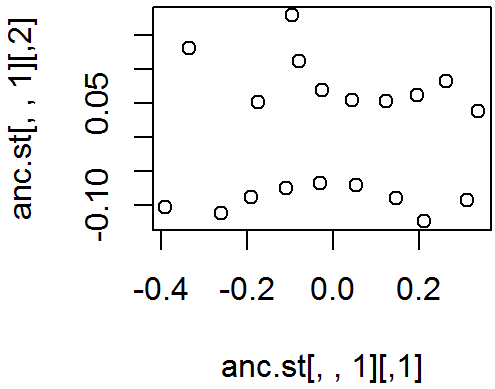
```
#Plot a phylomorphospace
phy.tar<-plotGMPhyloMorphoSpace(tree, tarsalsdf$coords, tip.labels =
```

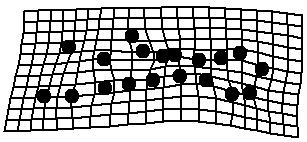
```
F, ancStates = T, plot.param = list(t.bg=palette()[tarsalsdf$family]))
```



Geomorph will also calculate the ancestral state shapes at the nodes

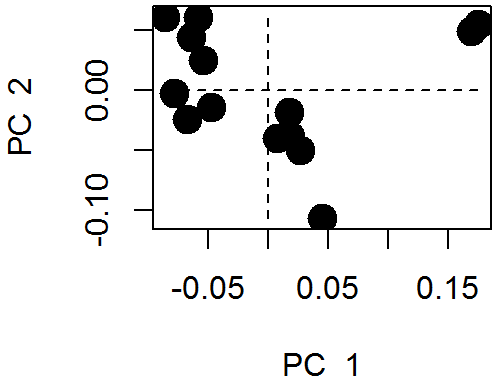
```
#Now we can also visualize ancestral states
anc.st<-arrayspecs(phy.tar, nrow(tarsalsdf$coords), 2)
plot(anc.st[,,1])
#Evolution along branch
plotRefToTarget(anc.st[,,1],anc.st[,,2], mag=3)
```





You can also use phytools to make a 3D phylomorphospace

```
a<-plotTangentSpace(tarsalsdf$coords,warpgrids = F,axis1=1,axis2=2,
verbose=F)
```




```
open3d()
#> wgl
#>   5
plotdat<-a$pc.scores[,1:3]
colnames(plotdat)<-c("","","")#prevent axis labels
obj<-phytools::phylomorphospace3d(tree,plotdat, method="dynamic",
                                  control=list(ftype="off",spin=FALSE,
box=FALSE), cex.symbol=0.5)

##Color by groups
spheres3d(a$pc.scores[,1:3], color=palette()[tarsalsdf$family],
r=0.01)
```

```
bbox3d(color = c("white"),shininess=15, alpha=0.3,xat=c(10),
xlab="",yat=c(10), ylab="",zat=c(10), zlab="")
text3d((a$pc.scores[,1:3]+0.02), texts =
substr(tarsalsdf$species,1,3))
```
You must enable Javascript to view this page properly.

Phylogenetic analyses

Geomorph uses Blomberg's K to test for strength and significance of phylogenetic signal.

```
#Test for phylogenetic signal
physignal(tarsalsdf$coords, tree, print.progress = F)
#>
#> Call:
#> physignal(A = tarsalsdf$coords, phy = tree, print.progress = F)
#>
#>
#>
#> Observed Phylogenetic Signal (K): 0.4906
#>
#> P-value: 0.002
#>
#> Based on 1000 random permutations
```

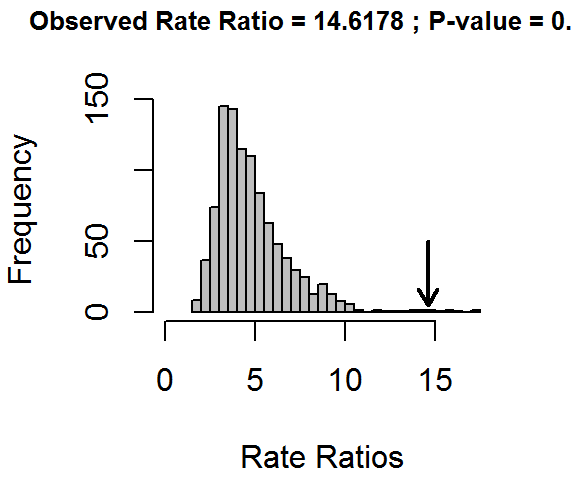
Now let's try a PGLS analysis in geomorph.

```
##Phylogenetic generalized least squares
#stance
pgls.tar<-procD.pgls(coords~stance, phy=tree, data=tarsalsdf,
print.progress = F)
pgls.tar$aov.table
#>           Df        SS         MS     Rsq      F       Z Pr(>F)
#> stance     2 0.0006130 0.00030652 0.07026 0.4156 -2.1816  0.971
#> Residuals 11 0.0081119 0.00073745 0.92974
#> Total     13 0.0087250

#size and stance with interaction
pgls.tar<-procD.pgls(coords~log(mass)*stance, phy=tree,
data=tarsalsdf,print.progress = F)
pgls.tar$aov.table
#>                  Df        SS         MS     Rsq      F        Z
Pr(>F)
#> log(mass)         1 0.0008420 0.00084203 0.09651 1.0749  0.35051
0.382
#> stance            2 0.0006296 0.00031478 0.07216 0.4019 -1.66532
0.938
#> log(mass):stance  2 0.0009868 0.00049338 0.11310 0.6299 -0.39641
0.660
#> Residuals         8 0.0062666 0.00078333 0.71824
```

```
#> Total             13 0.0087250
```

Finally, we can compare evolutionary rates in different portions of the tree based on brownian motion.

```
#Compare rates of evolution between families
names(tarsalsdf$family)<-tarsalsdf$species
rate.comp<-compare.evol.rates(tarsalsdf$coords, tree,
gp=tarsalsdf$family, print.progress = F)

plot(rate.comp)
rate.comp$sigma.d.gp
#>      Canidae      Felidae   Mephitidae   Mustelidae  Procyonidae
#> 1.044737e-05 1.165668e-05 8.224185e-06 1.063307e-05 3.840590e-06
#>      Ursidae
#> 5.713904e-05
rate.comp$pairwise.pvalue
#>             Canidae Felidae Mephitidae Mustelidae Procyonidae
#> Felidae       0.808
#> Mephitidae    0.920   0.928
#> Mustelidae    0.978   0.846      0.933
#> Procyonidae   0.399   0.377      0.185      0.458
#> Ursidae       0.051   0.072      0.001      0.077       0.001
```



Practical - PCM on Tarsals dataset

Now try applying some phylogenetic analyses to the tarsals dataset. How does phylogenetic correction change your results? How could you improve sampling to increase power in your analysis?

Take-home messages

1. The selection of landmarks ultimately determines results patterns, so think about this step very carefully!
2. Use all PCs in statistical tests (except CVA)
3. Use covariance methods instead of correlation in PCA
4. Always use non-parametric statistical tests where possible
5. GMM analyzes the entire shape, not what happens at just one landmark

Acknowledge and cite packages

When publishing your analyses, please always remember to cite packages that you are using in your scripts.

```
citation('geomorph')
#>
#> To cite package 'geomorph' in a publication use:
#>
#>   Adams, D. C., M. L. Collyer, and A. Kaliontzopoulou. 2018.
#>   Geomorph: Software for geometric morphometric analyses. R
#>   package version 3.0.6.
#>   https://cran.r-project.org/package=geomorph.
#>
#> A BibTeX entry for LaTeX users is
#>
#>   @Misc{,
#>     title = {Geomorph: Software for geometric morphometric
analyses. R package version 3.0.6.
https://cran.r-project.org/package=geomorph.},
#>     author = {D.C. Adams and M.L. Collyer and A. Kaliontzopoulou},
#>     year = {2018},
#>   }
#>
#> As geomorph is evolving quickly, you may want to cite also its
#> version number (found with 'library(help = geomorph)').
citation('StereoMorph')
#>
#> To cite StereoMorph in a publication use:
#>
#>   Olsen Aaron M, Mark W Westneat. 2015. StereoMorph: an R package
#>   for the collection of 3D landmarks and curves using a stereo
#>   camera set-up. Methods in Ecology and Evolution 6:351-356.
#>
#>   Olsen, AM & A Haber (2017). StereoMorph: Stereo Camera
#>   Calibration and Reconstruction. Version 1.6.1.
#>   https://CRAN.R-project.org/package=StereoMorph.
```

```
#>
#> To see these entries in BibTeX format, use 'print(<citation>,
#> bibtex=TRUE)', 'toBibtex(.)', or set
#> 'options(citation.bibtex.max=999)'.
```

Our Sponsors: *National Science Foundation (Sedimentary Geology and Paleobiology Program)*, *National Science Foundation (Earth Rates Initiative)*, *Paleontological Society*, *Society of Vertebrate Paleontology*