Landmark Data Collection

Ryan N. Felice

Load Packages

```
library(StereoMorph)
library(geomorph)
library(Morpho)
library(remotes)
library(Rvcg)
```

Collecting landmark data

Locate your files

One way to stay organised is to tell your R session the location of your "working directory", within which you have all of the files and folders you are working with. Note that the syntax for file paths is a little bit different on Mac and Windows.

On Mac:

```
setwd("/Users/Ryan/workshop/")
```

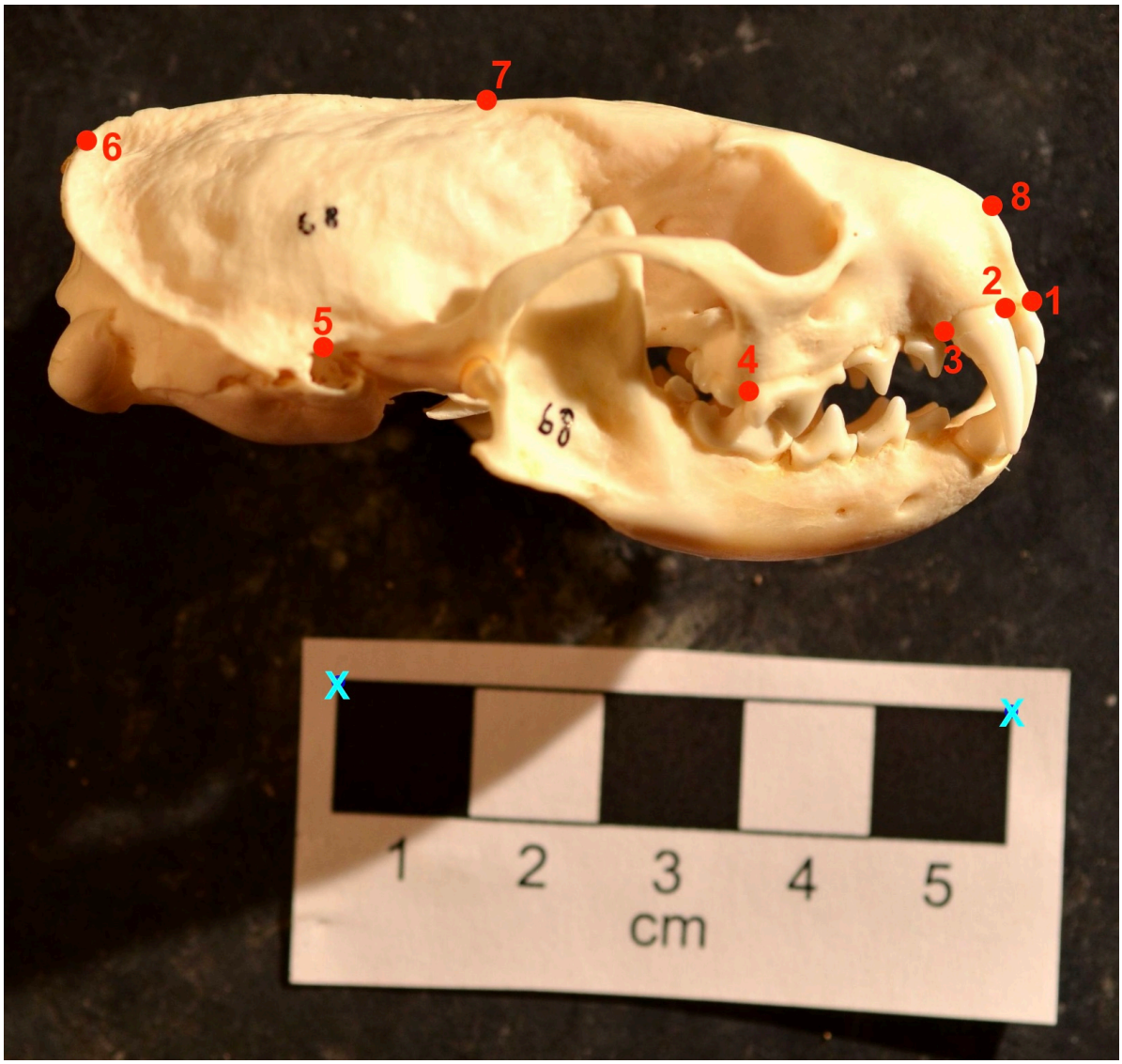
on windows

```
setwd("C:\\Users\\Ryan\\workshop\\")
```

Using Stereomorph to digitize images

digitizeImages includes three important inputs: image.file, the file path where you photos are located; shapes.file, the file path where you would like the coordinate data to be saved, and landmarks.ref, which is a simple txt file listing the names of your landmark locations in a single column. then you are ready start landmarking

```
digitizeImages(image.file = "./Mustelidae_skulls", #folder with images
               shapes.file = "./raw_landmarks", #target folder to store landmark data
               landmarks.ref = "./Data/landmark_scheme.txt") #text file with landmark definit
```



Keyboard shortcuts:

x = doubleclick = digitize point

p = previous point

n = next point

d = delete point

Load the landmark data back in to R

StereoMorph stores data as .txt files in a proprietary xml-like format. Luckily they provide a useful function to load these files into R.

```
raw_data<-readShapes(file = "./raw_landmarks")
```

Lets explore this data! What format is it in? It's a named list containing "ruler.pixel", "ruler.interval", "scaling.units" "scaling" "landmarks.pixel" "landmarks.scaled" and "ruler.points" for each specimen.

You can explore each by using the dollar sign ($) to call that named item in the list. for example:

```
### number of pixels between
#the two points you digitized on the rulers
raw_data$ruler.pixel
```

```
    MartesFoina_1_lateral     MartesFoina_2_lateral     MartesFoina_3_lateral
                 1466.136                  1478.221                  1473.391
     MelesMeles_1_lateral      MelesMeles_2_lateral      MelesMeles_3_lateral
                 1251.392                  1252.032                  1237.026
 MustelaErminea_1_lateral  MustelaErminea_2_lateral  MustelaErminea_3_lateral
                 1467.077                  1468.417                  1461.575
MustelaLutreola_1_lateral MustelaLutreola_2_lateral MustelaLutreola_3_lateral
                 1469.466                  1475.570                  1481.540
 MustelaNivalis_1_lateral  MustelaNivalis_2_lateral  MustelaNivalis_3_lateral
                 2300.498                  2297.642                  2290.022
```

But what we are really interested is `landmarks.scaled`. this is the raw XY coordinate data scaled based on the rulers we defined. What format is this data?

```
class(raw_data$landmarks.scaled)
```

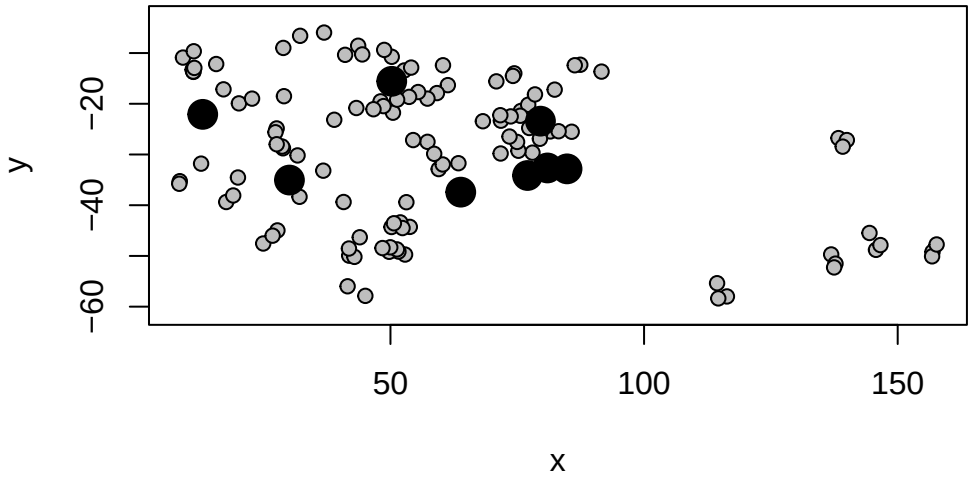
```
[1] "array"
```

```
 ### the raw landmark coordinates, scaled
dim(raw_data$landmarks.scaled)
```

```
[1]  8  2 15
```

its an "array" with three "dimensions". The first dimension (8) is the number of landmarks. The second (2) is the number of dimensions of the data (2D data, x and y coordinates = 2) and the third dimension is the number of specimens. first, lets look at our data all together

```
plotAllSpecimens(raw_data$landmarks.scaled)
```



It doesnt look very nice yet because the specimens are all at different positions and rotations. Lets look at one specimen on it's own. You can manipulate a 3d array using R's notation within square brackets. for 3D arrays of landmark data, the notation is array[landmarks, dimensions, specimens]. Leaving a space blank in this notation means you want to include everything. Try it out:

we might also want to subset the data based on specimen names. to access the specimen names we can use the dimnames function

```
dimnames(raw_data$landmarks.scaled)
```

```
[[1]]
[1] "Anterioventral point on premaxilla, contact with incisor"
[2] "Anterior border of the C1 alveolus"
[3] "Posterior border of the C1 alveolus"
[4] "Anterior border of the P4 alveolus"
[5] "Superior border of the external auditory meatus"
[6] "Posterior most point on the sagittal crest"
[7] "Anterior most point on the sagittal crest"
[8] "Dorsal point on external naris"

[[2]]
NULL

[[3]]
 [1] "MartesFoina_1_lateral"     "MartesFoina_2_lateral"
 [3] "MartesFoina_3_lateral"     "MelesMeles_1_lateral"
 [5] "MelesMeles_2_lateral"      "MelesMeles_3_lateral"
 [7] "MustelaErminea_1_lateral"  "MustelaErminea_2_lateral"
 [9] "MustelaErminea_3_lateral"  "MustelaLutreola_1_lateral"
[11] "MustelaLutreola_2_lateral" "MustelaLutreola_3_lateral"
[13] "MustelaNivalis_1_lateral"  "MustelaNivalis_2_lateral"
[15] "MustelaNivalis_3_lateral"
```

Note that Stereomorph helpfully stores names for eah landmark based on your landmarks.ref file, but not all landmarking software does this. For now, lets subset the landmark data to be only the specimen called "MustelaLutreola_2_lateral"

```
raw_data$landmarks.scaled[,,which(dimnames(raw_data$landmarks.scaled)[[3]]
                                  == "MustelaLutreola_2_lateral")]
```

```
                                                      [,1]      [,2]
Anterioventral point on premaxilla, contact with incisor 85.72960 -25.51557
Anterior border of the C1 alveolus                        83.15432 -25.41391
Posterior border of the C1 alveolus                       79.46083 -26.90486
Anterior border of the P4 alveolus                        63.39924 -31.71656
Superior border of the external auditory meatus           28.63301 -28.42970
Posterior most point on the sagittal crest                11.11435 -13.65574
Anterior most point on the sagittal crest                 50.25178 -10.74161
Dorsal point on external naris                            82.37496 -17.21369
```

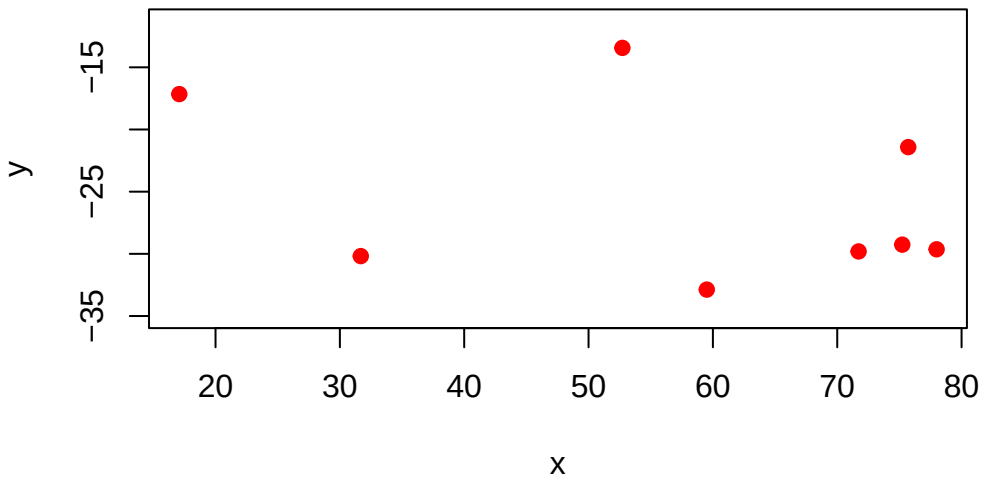
similarly, we could select just the specimens whose names contain the word “Nivalis”

```
Nivalis <- raw_data$landmarks.scaled[,,grep("Nivalis",
                                    dimnames(raw_data$landmarks.scaled)[[3]])]
Nivalis
```

```
, , MustelaNivalis_1_lateral

                                                      [,1]      [,2]
Anterioventral point on premaxilla, contact with incisor 53.81444 -44.27303
Anterior border of the C1 alveolus                    51.94528 -43.38192
Posterior border of the C1 alveolus                   50.20653 -44.29476
Anterior border of the P4 alveolus                    43.94701 -46.31606
Superior border of the external auditory meatus       27.71139 -44.96853
Posterior most point on the sagittal crest            19.93047 -34.53600
Anterior most point on the sagittal crest             36.75292 -33.18847
Dorsal point on external naris                        53.14067 -39.40452

, , MustelaNivalis_2_lateral

                                                      [,1]      [,2]
Anterioventral point on premaxilla, contact with incisor 52.88031 -49.72490
Anterior border of the C1 alveolus                    51.57462 -49.13734
Posterior border of the C1 alveolus                   49.74666 -49.18086
Anterior border of the P4 alveolus                    42.89181 -50.18189
Superior border of the external auditory meatus       24.91685 -47.54876
Posterior most point on the sagittal crest            17.62677 -39.36645
Anterior most point on the sagittal crest             40.75919 -39.36645
Dorsal point on external naris                        52.37980 -44.50215

, , MustelaNivalis_3_lateral

                                                      [,1]      [,2]
Anterioventral point on premaxilla, contact with incisor 51.26589 -48.71133
Anterior border of the C1 alveolus                    49.95586 -48.29648
Posterior border of the C1 alveolus                   48.42749 -48.44932
Anterior border of the P4 alveolus                    41.81183 -48.53666
Superior border of the external auditory meatus       26.76830 -46.00393
Posterior most point on the sagittal crest            18.99545 -38.07824
Anterior most point on the sagittal crest             32.07393 -38.34025
Dorsal point on external naris                        50.67637 -43.55854
```

Plot one specimen

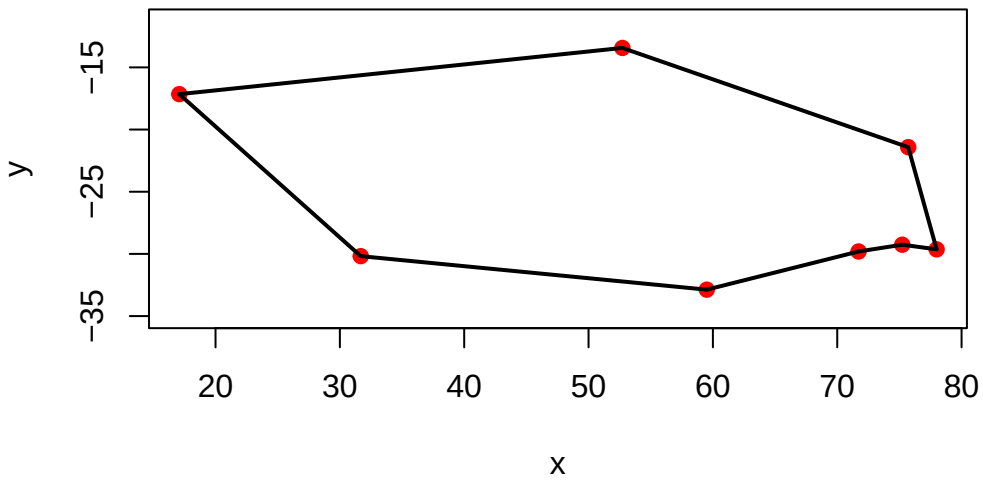
```
plot(raw_data$landmarks.scaled[,,1],asp = 1,
     pch = 19,col="red",
     xlab = "x", ylab = "y")
```



Not much to look at! it would make it nicer if we could see lines connecting the points. With the geomorph package we can define "links" to join up the points.

Now plot with the links!

```
my_links <- read.csv("./Data/links.csv")[,2:3]

specimen <- raw_data$landmarks.scaled[,,1]
plot(specimen,asp = 1, pch = 19,col="red",
     xlab = "x", ylab = "y")
links<-my_links
for (i in 1:nrow(links)) {
                  segments(specimen[links[i, 1], 1],
                           specimen[links[i, 1], 2],
                           specimen[links[i, 2], 1],
                           specimen[links[i, 2], 2],
                    col = "black", lty = 1, lwd = 2)
                }
```



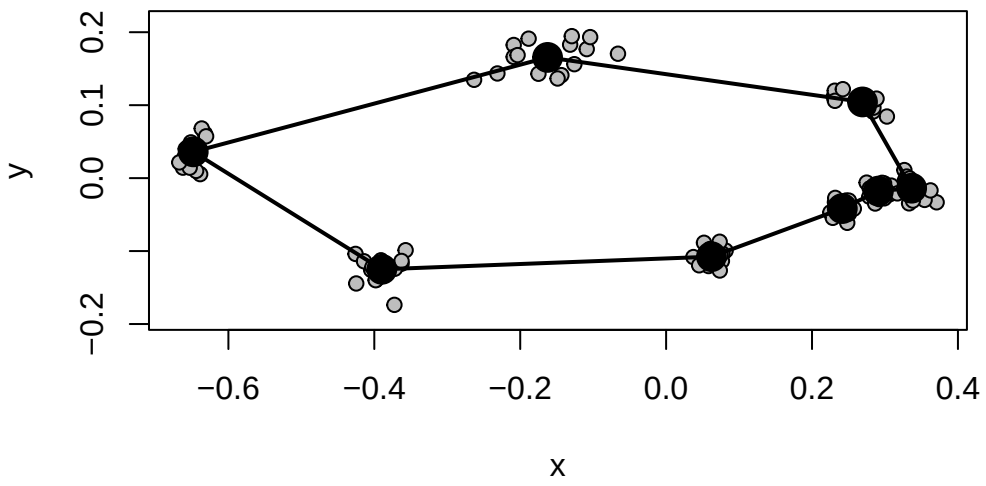
Procrustes alignment

```
Y.gpa<- gpagen(raw_data$landmarks.scaled)
```

```
Performing GPA

  |
  |                                                                      |   0%
  |
  |==================                                                    |  25%
  |
  |===================================                                   |  50%
  |
  |======================================================================| 100%

Making projections... Finished!
```

now lets explore the output. Y.gpa contains a lot of stuff, but the most important are the alligned landmarks (Y.gpa$coords) and the centroid size of each specimen (Y.gpa$Csize)

```
plotAllSpecimens(Y.gpa$coords, links = my_links)
```

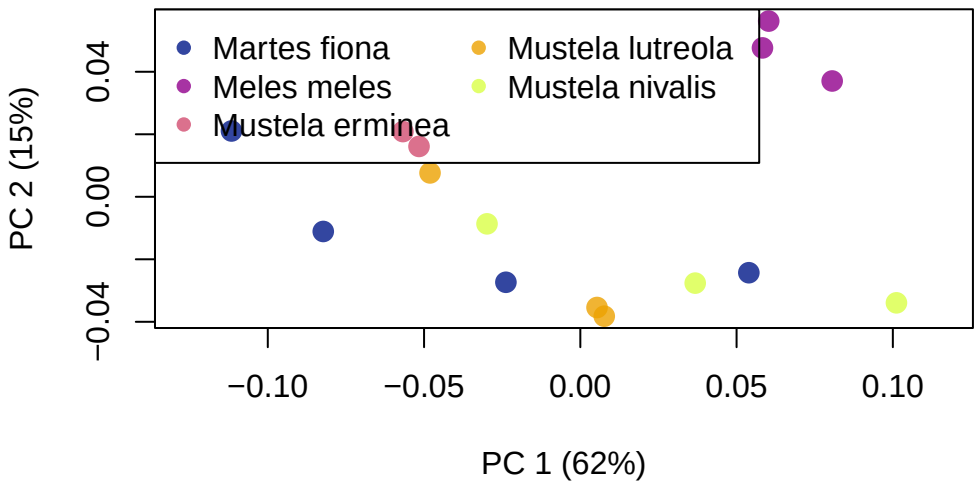


Plot your data

A common way to explore your geometric morphometric data is principal component analysis (PCA). Lets take a look.

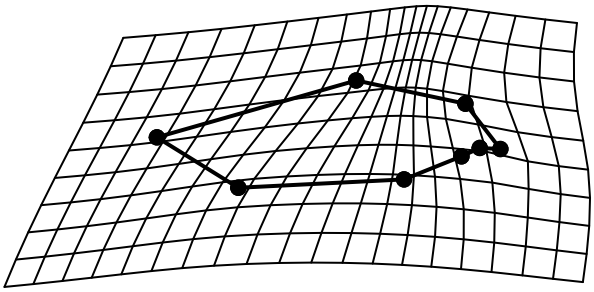
```
my_PCA <- gm.prcomp(Y.gpa$coords)
species_IDs <- c("Martes_fiona",
                 "Martes_fiona",
                 "Martes_fiona",
                 "Meles_meles",
                 "Meles_meles",
                 "Meles_meles",
                 "Mustela_erminea",
                 "Mustela_erminea",
                 "Mustela_lutreola",
                 "Mustela_lutreola",
                 "Mustela_lutreola",
                 "Mustela_nivalis",
                 "Mustela_nivalis",
```

```
                  "Mustela_nivalis")
species_IDs <- as.factor(species_IDs)

#Calculate proportion of variance explained per axis
PoV<-(my_PCA$sdev^2/sum(my_PCA$sdev^2))*100
#Color-code by specimen
cols<-hcl.colors(n=length(unique(species_IDs)),
                 palette="plasma", alpha=0.8)
Col.vec<-cols[species_IDs]
#Plot results
plot(my_PCA$x[,1], my_PCA$x[,2], type="p",  asp=1,
     xlab=paste("PC 1 (", round(PoV[1], digits=0), "%)",
                sep=""),
     ylab=paste("PC 2 (", round(PoV[2], digits=0), "%)",
                sep=""),
     pch=16, cex=1.5, col=Col.vec)
legend("topleft", pch=16, col=cols,
       legend = c("Martes fiona",
                  "Meles meles",
                  "Mustela erminea",
                  "Mustela lutreola",
                  "Mustela nivalis"), ncol=2)
```



lets visualize the shape change across PC Axis 1 using TPS (thin plate spline)

```
plotRefToTarget(M1 = my_PCA$shapes$shapes.comp1$min,
                M2 = my_PCA$shapes$shapes.comp1$max,
                method = "TPS",
                links = my_links)
```



How can we interpret this shape change?

Some 3D shape visualization

When we collect high-density 3d Morphometric data, there is a risk that we have too many landmarks. One way to check if you have collected too many landmarks is using LaSEC. This method calculates how much variation is in your dataset, then starts removing landmarks randomly and recalculating variance to see how many landmarks you need to capture most of the information that you are interested in.

To use this package, you need to install it from Github

```
remotes::install_github("akiopteryx/lambda")
library(LaMBDA)
```

Load in the example 3D data from geomorph

```
data("scallops")
scallop.gpa <- gpagen(A = scallops$coorddata,
                      curves = scallops$curvslide,
                      surfaces = scallops$surfslide)
```

Run the analysis. By default, this will use 1000 iterations, but that is a little slow so let's use 100 iterations.

```
lasec_test <- lasec(coord.data = two.d.array(scallops$coorddata),
                    n.dim = 3,
                    iter=100)
```

Now we can plot the results

```
plot.lasec(lasec_test)
abline(v=min(which(lasec_test$median.fit>=0.99)),
       col="red", lwd=2)
```

How many landmarks should we retain?

How should we remove unneeded landmarks?

Plotting shape differences in 3D

The TPS spline we made for the mustelid skulls are a bit hard to think about or visualze in 3D. How can we visualize shape change in 3D? First let's load in some example data

```
raw_mammals_3d<-read.csv("./Data/mammals.csv",
                         row.names = 1)
raw_mammals_3d
```

We have imported this as a csv file. We need to convert this two a 2D array to make it compatible with geomorph and other shape analysis packages

```
mammals_3d <- arrayspecs(A = raw_mammals_3d,
                         p = ncol(raw_mammals_3d)/3, k = 3)
```

The reverse of arrayspecs is two.d.array, which is really useful for converting your array back to a matrix that can be easily stored as a table:

```
raw_mammals_3d_2 <- two.d.array(mammals_3d)
#write.csv(raw_mammals_3d_2, file = "./Data/mammals2.csv")
```

We have 3 specimens here:

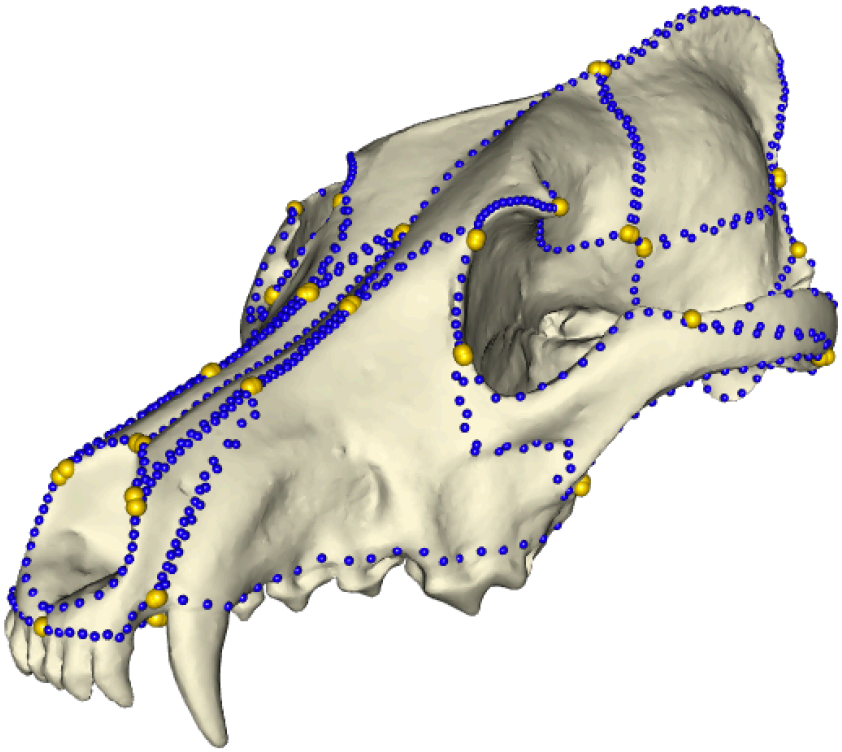
```
dim(mammals_3d)
dimnames(mammals_3d)
```

These are *Balaenoptera musculus* (blue whale), *Canis lupus* (wolf), and *Pan troglodytes* (chimpanzee). lets load in a 3d mesh of the wolf for easier visualization, as well as a list that tells R which landmarks are fixed points and which ones are semilandmarks.

```
canis_mesh <- vcgImport("./Data/Canis_lupus.ply")
lm_dat <- read.csv("./Data/mammal_3d_fixed_points.csv")[,1]
```

Visualize the mesh and the landmarks together. This will open a new window.

```
shade3d(canis_mesh, col = "#FFFADC", specular = "black")
spheres3d(mammals_3d[,,"Canis_lupus"],col="blue",radius=1)
spheres3d(mammals_3d[lm_dat,,"Canis_lupus"],col="gold",radius=2)
```



Now we can warp the shape of this *Canis lupus* mesh to match the landmark configuration of one of the other species.

```
wolf_to_whale <- tps3d(x = canis_mesh,
                       refmat = mammals_3d[,,"Canis_lupus"],
                       tarmat = mammals_3d[,,"Balaenoptera_musculus"])
#plot
open3d()
shade3d(wolf_to_whale, col = "#FFFADC", specular = "black")
```

This is the same way you would create mesh models of extreme shapes calculated from PCA

Try it yourself with the chimpanzee specimen. How would you visualize multiple mesh files in the same 3D window?