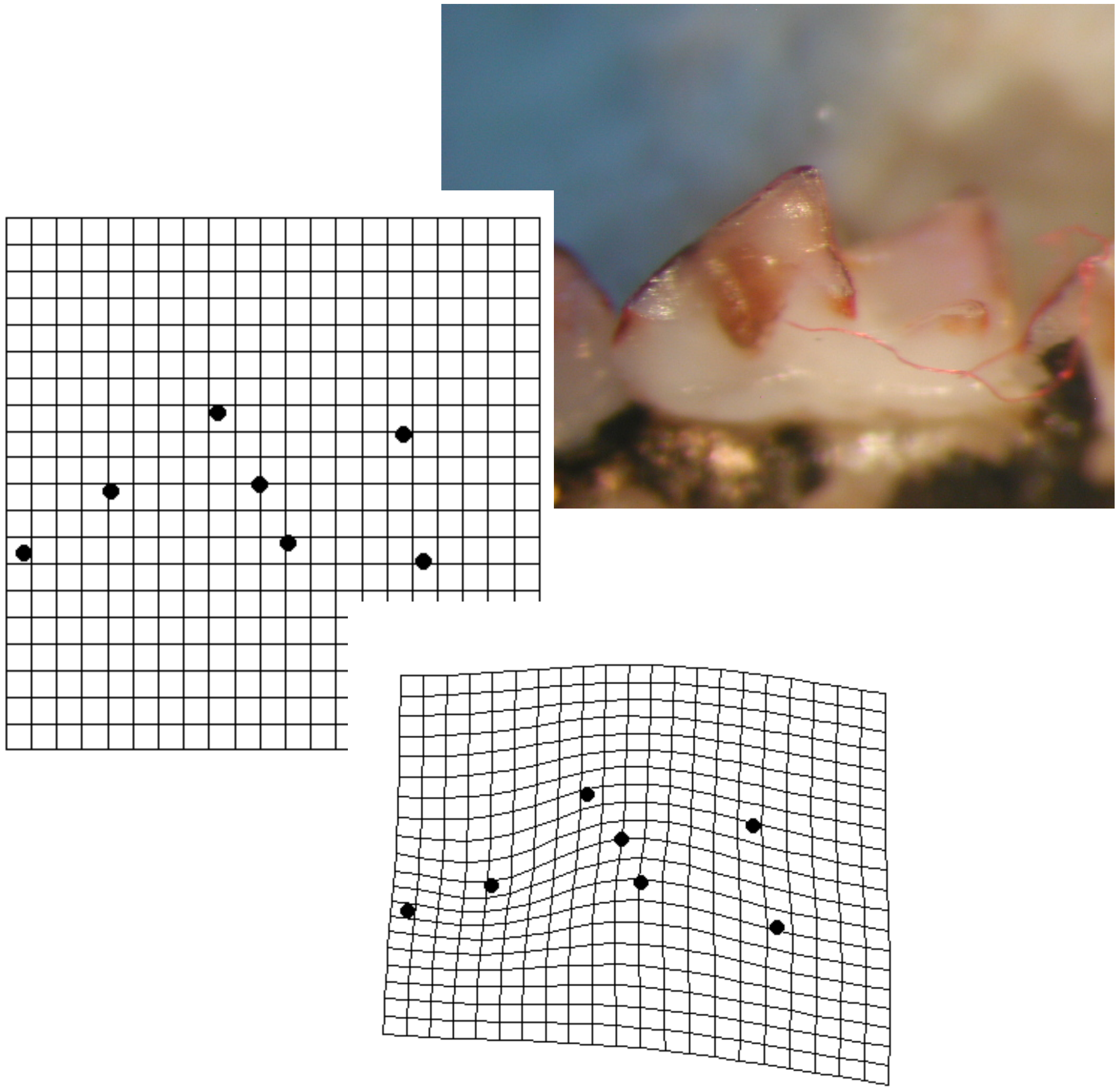
A basic geometric morphometric analysis

Step by step in *R*



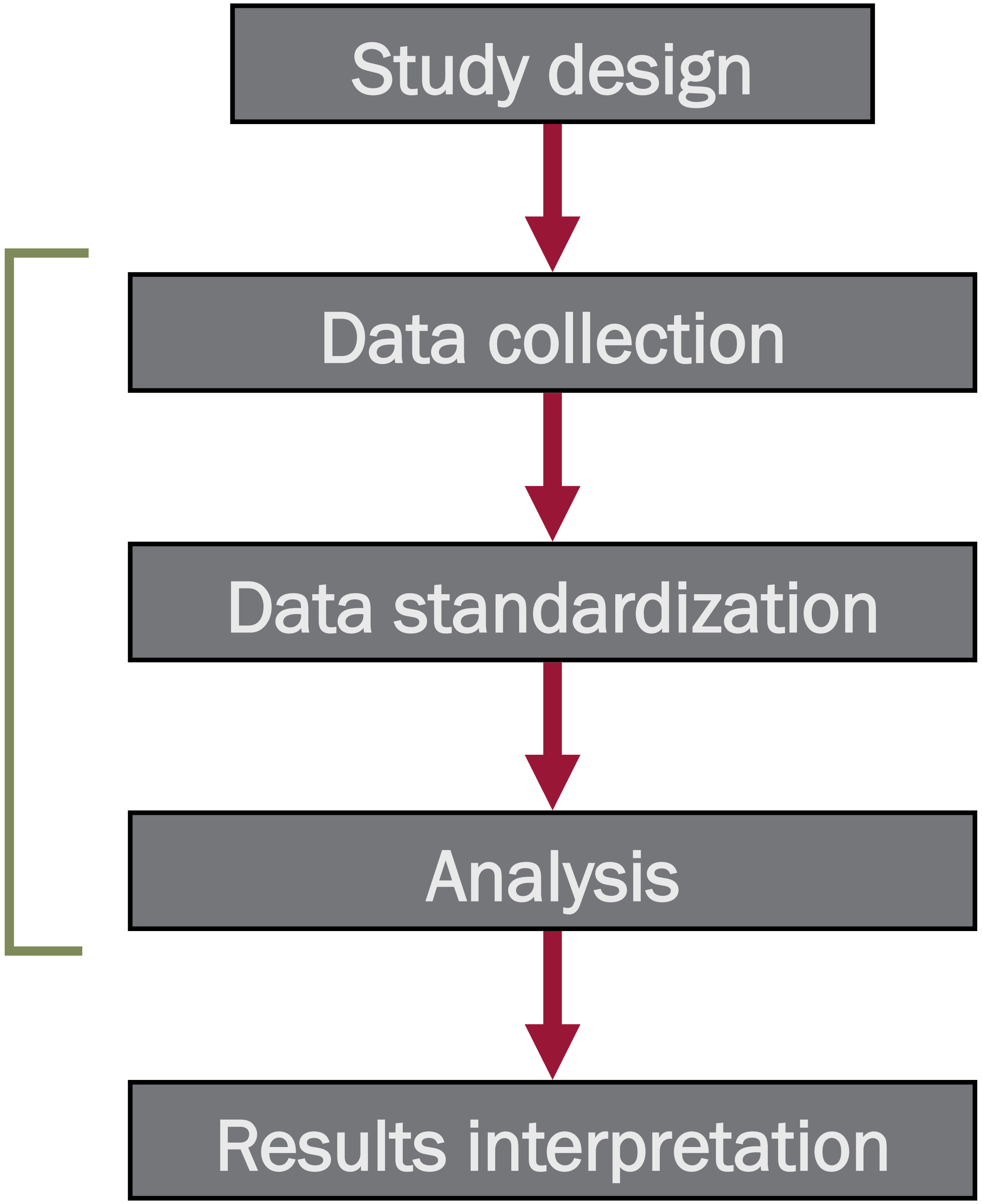
R packages to install

geomorph	*Geometric morphometrics package by Adams and Otárola-Castillo*
shapes	*Geometric morphometrics package by Ian Dryden*
svd	*Singular value decomposition package*
scatterplot3d	*Functions for 3D plotting (installed as dependency to above)*
rgl	*More 3D functions (installed as dependency to above)*
MASS	*Modern Applied Statistics with S (installed as dependency to above)*
ape	*Analyses of Phylogenetics and Evolution (installed as dependency to above)*
vegan	*Community ecology package (installed as dependency to above)*

Example file: Bialowieza_Bialowieza.tps

Install from CRAN
Load with library(*packagename*)

General Procedure



Study design

Data collection

Data standardization

Analysis

Results interpretation

Steps in a geometric morphometric analysis

1. Obtain landmark coordinates.

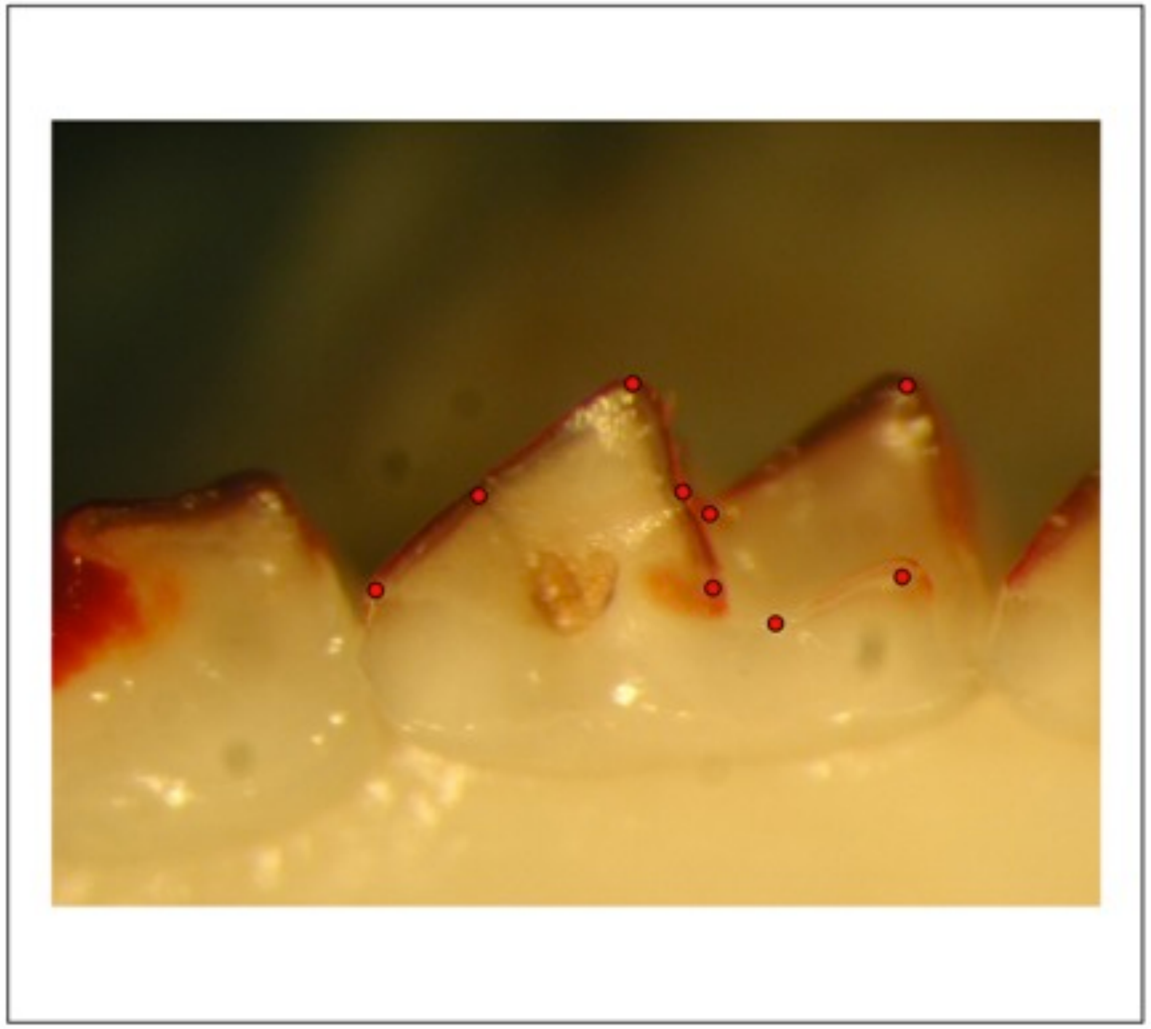
2. Standarization: Procrustes superimposition of the landmarks.
   removes size, translation, and rotation
   (also project to tangent space for further statistical analysis).

3. Standardization: Create shape variables
   (variables that retain information about shape, but are uncorrelated with one another and have appropriate degrees of freedom)
   PCA scores are one kind of shape variable that satisfy these requirements

4. Analysis: Conduct further analyses on the shape variables
   Type of analysis required depends on questions being addressed
   Regression, ANOVA, path analysis, tree-building, etc.

Choosing landmarks

1. landmarks should sample the aspects of the shape that are of interest
2. enough landmarks should be used to adequately sample the shape
3. don’t oversample (NB: each landmark adds weight to the analysis. Multiple landmarks on one area will increase weight of that area.)
4. landmarks should be repeatable (same point on every specimen, placed with as little error as possible)
    1. type 1 = location of the point defined by obvious biologically homologous structures (eg. intersection of three bones)
    2. type 2 = location of the point defined by obvious geometry (eg., point of greatest curvature)
    3. type 3 = location of point defined with reference to another point (eg., point ventral to last tooth)



5. landmarks must be placed in the same order on all specimens

Obtaining landmark coordinates

From a file

```
lands <- readland.tps(file.choose())    [imports from TPS format file]

lands <- readland.nts(file.choose())    [imports from NTS format file]
```

From an image

```
digitize2d(filename, landmarks, scale)    [opens jpg file and collects lands]
```

Note: digitize2d may need debugging. Saves coordinates to file in working directory in NTS format.

```
readmulti.nts(filenames)
```

digitize2d()

Erik Otarola-Castillo has sent a fix. Do the following and the function will work:

1. Set the working directory to where your images are stored.
2. Define the function picscale() as follows

```
picscale <- function(scale){
  digscale<-NULL
  digscale<-locator(2,type="o",lwd=2,col="red",lty="11")
  sqrt(sum(diff(digscale$x)^2 + diff(digscale$y)^2))*scale
}
```

To collection landmarks in digitize2d:

1. If you have a scale bar in the images

   1.1. enter the length of the scale bar as argument (e.g., 10 if the scale bar is 10 mm)
   1.2. when digitize2d starts, click on both ends of the scale bar, then click on your landmarks in the proper order
   1.3. coordinates and centroid sizes will be scaled in the units of your scale bar (e.g., mm)

2. If you don't have a scale bar:

   2.1. enter 1 as the scale argument
   2.2. when digitize2d starts, click in two arbitrary but different places on the image, then click on your landmarks in the proper order
   2.3. coordinates will have no real units and centroid size will be meaningless

Automate the digitizing and loading of files

```
Digitize.My.Files <- edit()


function(path, lands, scale) {
setwd(path)
myFiles  <- dir(pattern="[jJ][pP][gG]")
for( i in 1:length( myFiles ) )   {
dig2d(myFiles[i],lands, scale)
}
myFiles <- dir(pattern="[nN][tT][sS]")
return(readmulti.nts(myFiles))
}



mylands <- Digitize.My.Files(“/Users/pdavidpolly/ShrewsAndMarmots”, 10, 10)
```

Cryptic error?

```
Error in .External2(C_edit, name, file, title, editor) :
  unexpected input occurred on line 3
 use a command like
 x <- edit()
 to recover
```

May be caused by smart quotes if you copied and pasted script from Word or other text editor

Performing Procrustes analysis and obtaining shape variables

Also known as Generalized Procrustes Analysis (GPA)

```
> gpa.lands <- gpagen(lands)
```

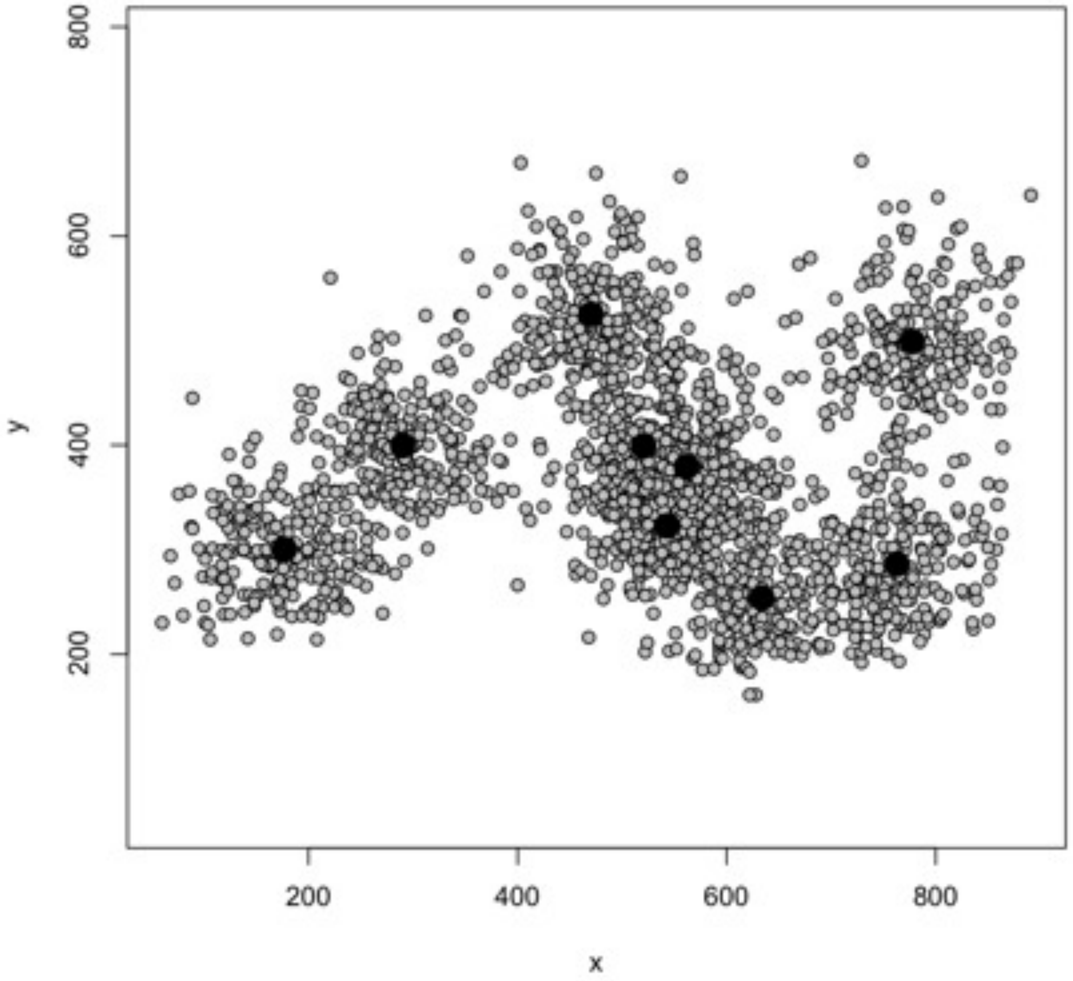
results of gpagen() include:

1. a plot of the superimposed specimens and their consensus (mean) shape
2. superimposed coordinates ($coords)
3. shape variables as principal components scores ($pc.scores)

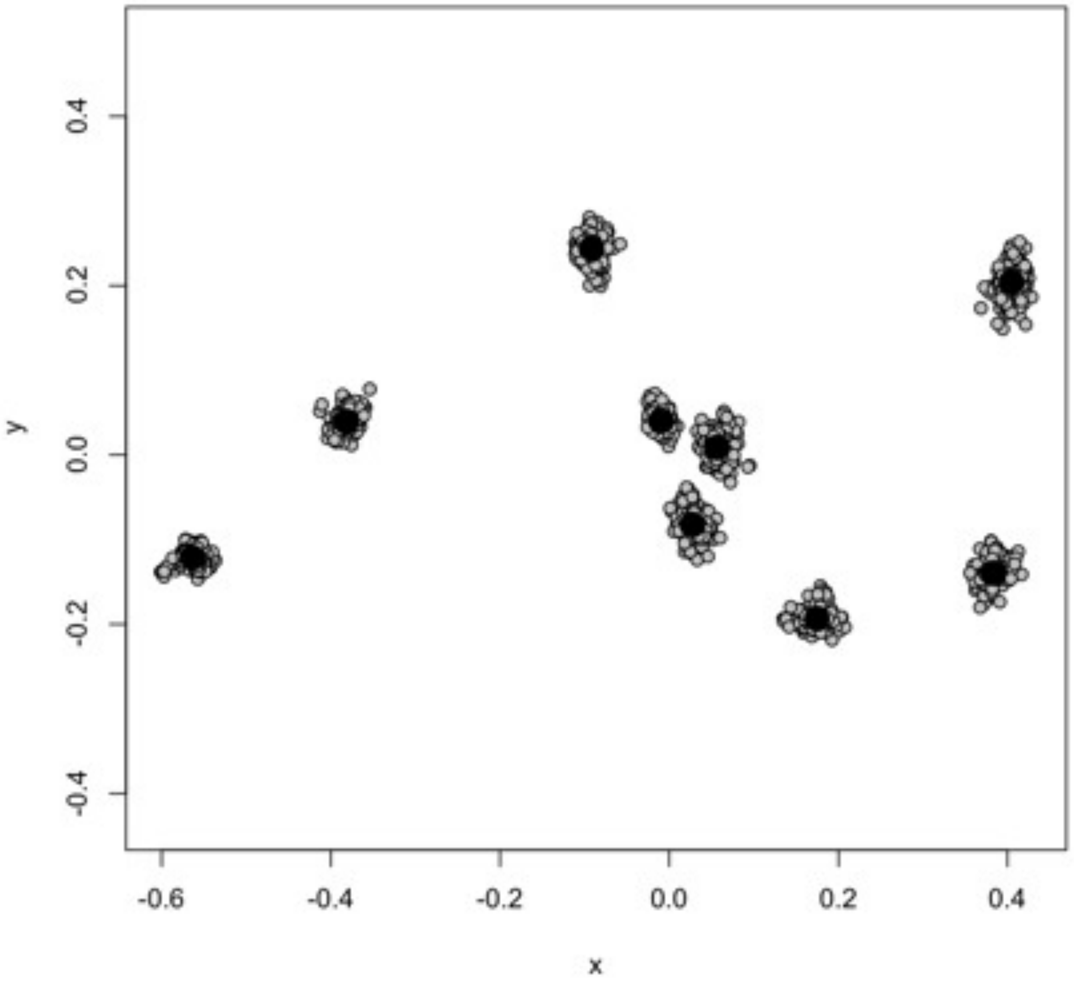
Procrustes of shrew teeth

data from “Bialowieza_Bialowieza.tps”

Before



After



> plotAllSpecimens(lands)

> gpagen(lands)

Details of *gpagen()*

- `proc <- gpagen(`
- This function does a generalized Procrustes analysis, superimposition of multiple specimens about their mean (Gower, 1975; Rohlf and Slice, 1990)
- Returns a plot of superimposed coordinates
- Returns the superimposed coordinates and their centroid sizes in dataframe format:
  - $coords = the x,y(,z) Procrustes coordinates after superimposition
  - $Csize = the centroid size of the specimens

To plot the Procrustes coordinates yourself:

```
for(i in 1:dim(proc$coords)[3]) points(proc$coords[,,i])

plotAllSpecimens(proc$coords)
```

Creating a principal components plot

```
> pca.lands <- plotTangentSpace(proc$coords, label=TRUE)
```

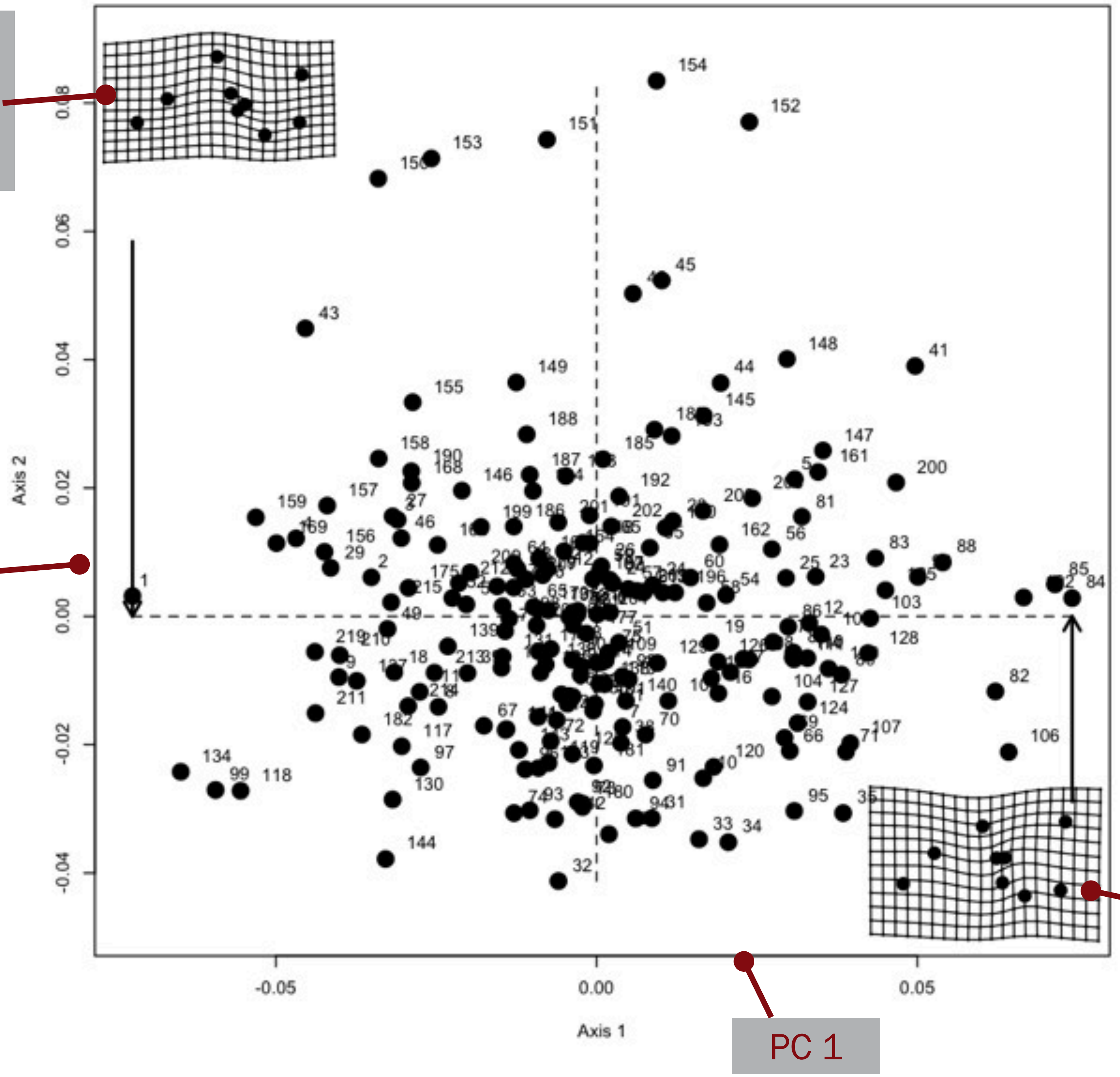
results of plotTangentSpace() include:

1. a plot of the specimens in principal components space and two thin plate spline grids showing the shapes associated with the positive and negative ends of the horizontal axis (default = PC1 x PC2   set axes using the args axis1 and axis2)

2. summary of variance associated with each PC axis ($pc.summary)

3. shape variables as principal components scores ($pc.scores)

Principal component plot of shrew teeth



Thin plate spline grid of shape at negative end of 1st axis

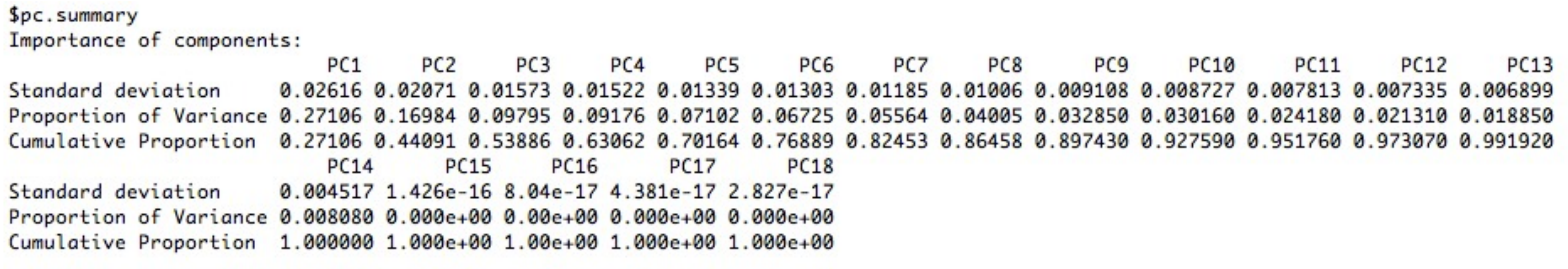
PC 2

Thin plate spline grid of shape at positive end of 1st axis

PC 1

Variance explained by PC axes

```
pca.lands$pc.summary
```



```
$pc.summary
Importance of components:
                           PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8      PC9     PC10     PC11     PC12     PC13
Standard deviation     0.02616 0.02071 0.01573 0.01522 0.01339 0.01303 0.01185 0.01006 0.009108 0.008727 0.007813 0.007335 0.006899
Proportion of Variance 0.27106 0.16984 0.09795 0.09176 0.07102 0.06725 0.05564 0.04005 0.032850 0.030160 0.024180 0.021310 0.018850
Cumulative Proportion  0.27106 0.44091 0.53886 0.63062 0.70164 0.76889 0.82453 0.86458 0.897430 0.927590 0.951760 0.973070 0.991920
                           PC14      PC15     PC16      PC17      PC18
Standard deviation     0.004517 1.426e-16 8.04e-17 4.381e-17 2.827e-17
Proportion of Variance 0.008080 0.000e+00 0.00e+00 0.000e+00 0.000e+00
Cumulative Proportion  1.000000 1.000e+00 1.00e+00 1.000e+00 1.000e+00
```

Important concept: variance of a data set

Variance = variability = shape variation

PC scores preserve the shape variation in the original data

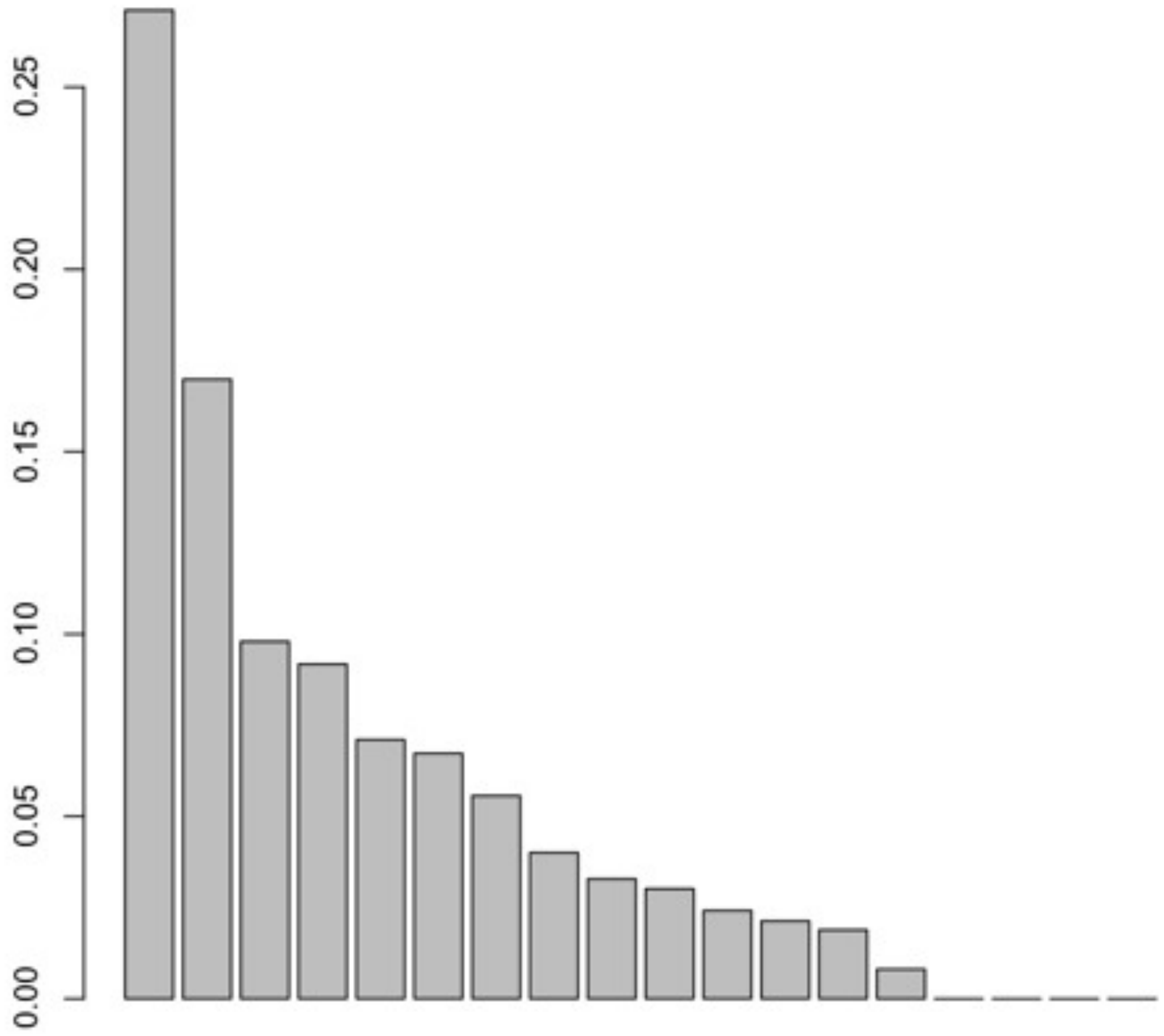
Variance = average squared distance from the mean

Variance in Procrustes coordinates = Variance in PC scores

```
> sum(pca.lands$pc.summary$sdev^2)

> sum(apply(pca.lands$pc.scores, 2, var))

> sum(apply(proc$coords,c(1,2),var))
```

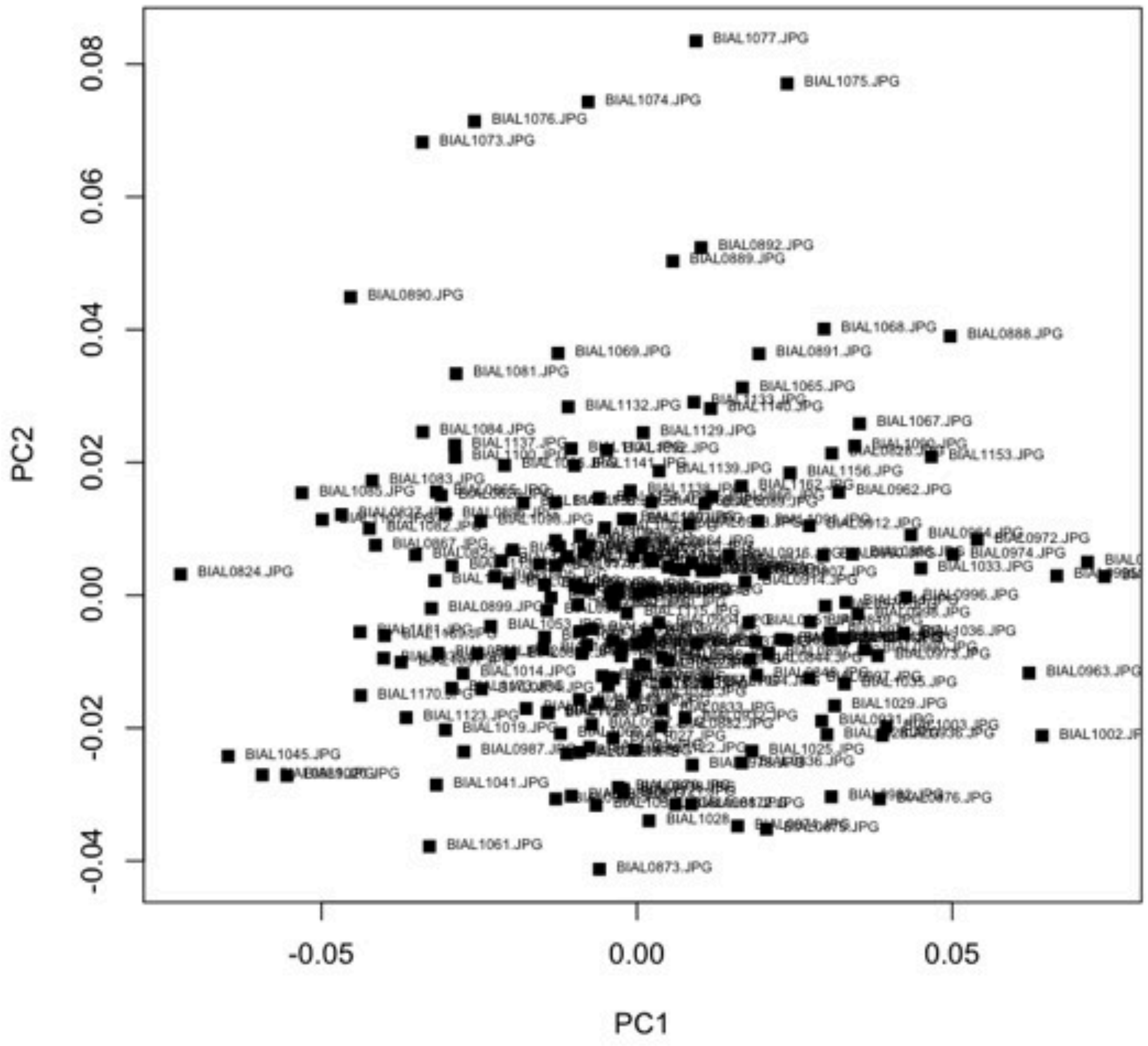
Show scree plot of variance explained

```
barplot(pca.lands$pc.summary$sdev^2/sum(pca.lands$pc.summary$sdev^2))
```



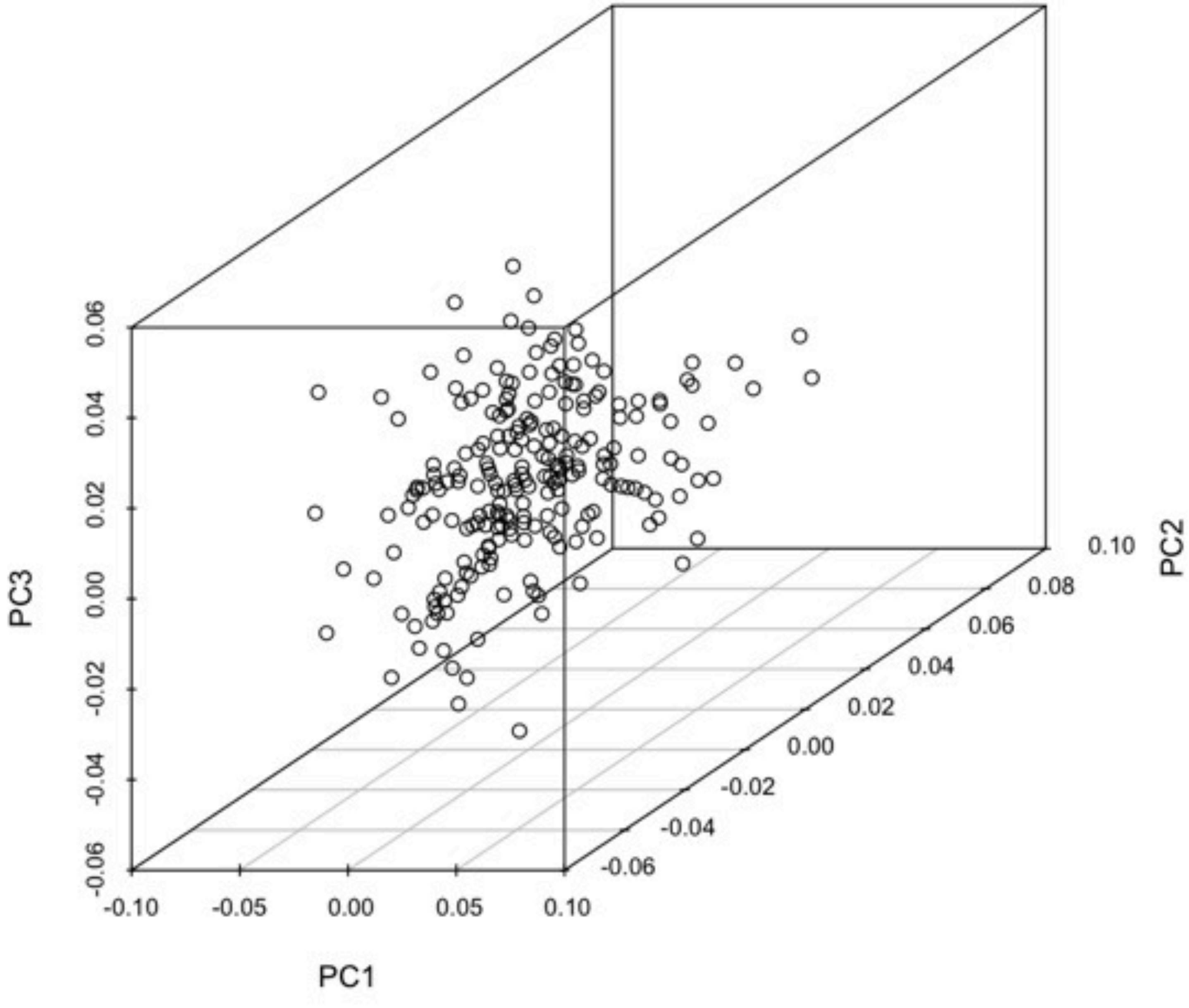
Create your own customized plot

```
> plot(pca.lands$pc.scores[,1:2],pch=15,xlab="PC1",ylab="PC2")

> text(pca.lands$pc.scores[,1:2],rownames(pca.lands$pc.scores[,1:2]),pos=4,cex=.5)
```



Create 3D scatterplot

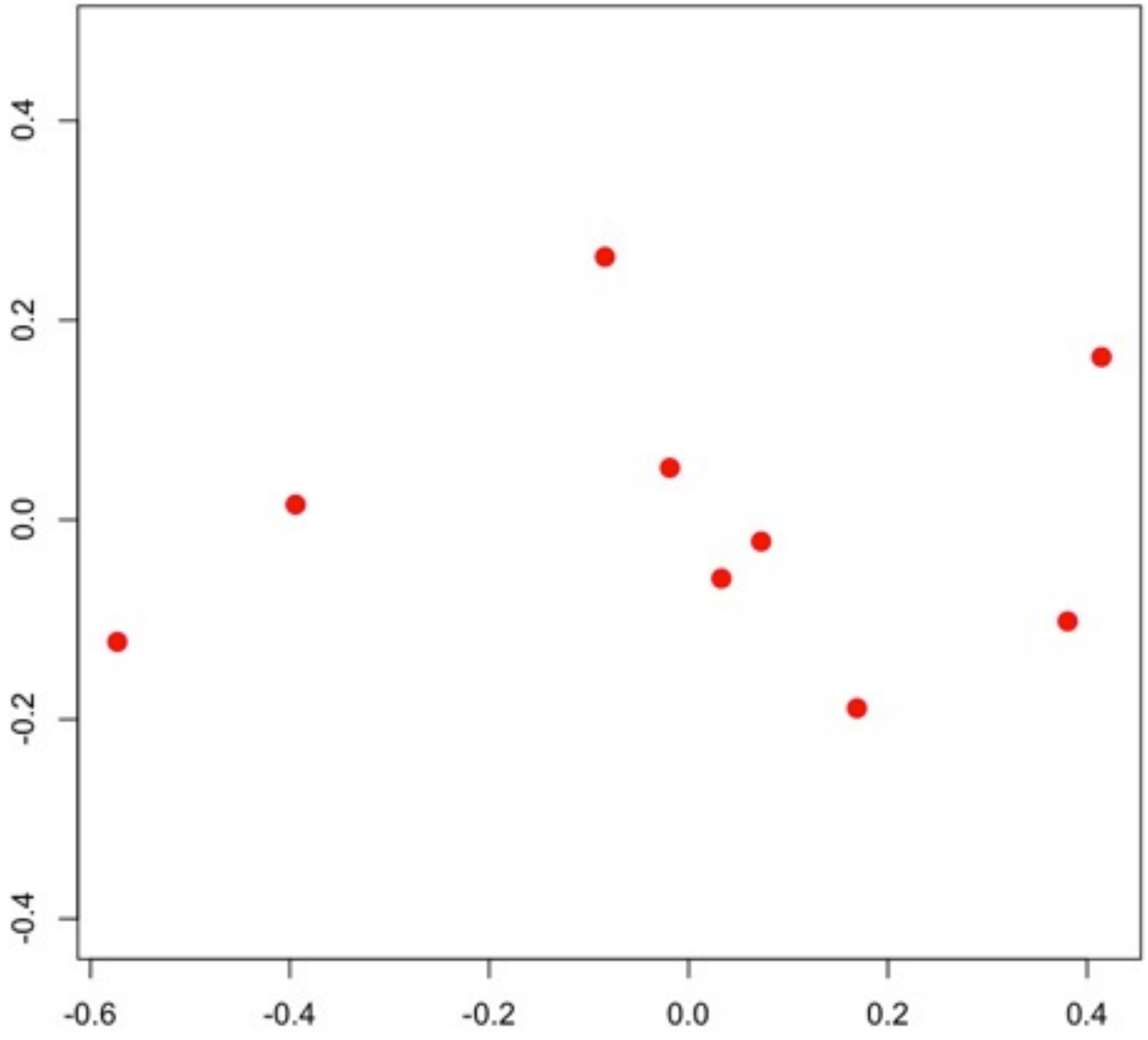
```
> library(scatterplot3d)

> scatterplot3d(pca.lands$pc.scores[,1:3])
```



Terminology: shape

Shape = a set of landmarks

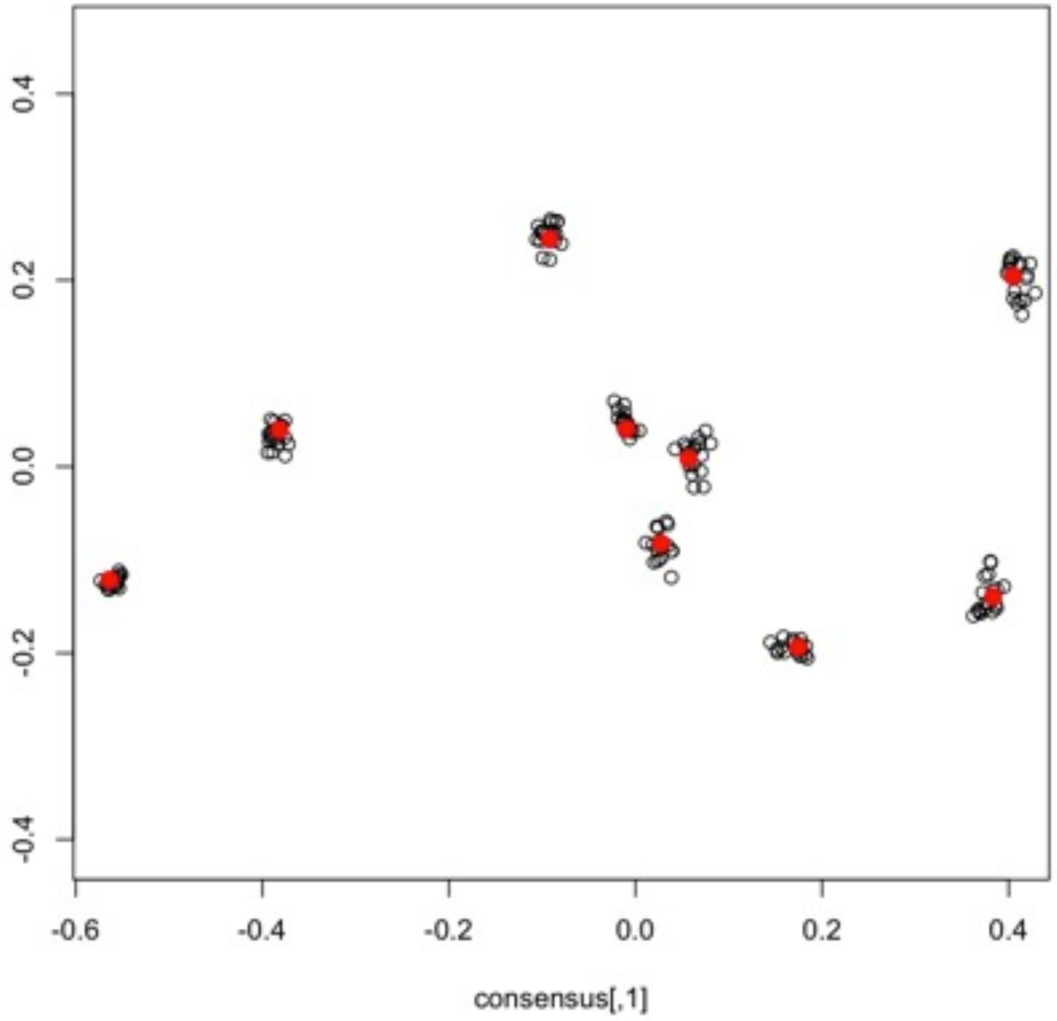
Sometimes "shape" specifically refers to the landmark configurations without respect to size and "form" refers to the configurations including size



Terminology: Consensus shape

Consensus shape = mean shape

Average x,y coordinate of each landmark after Procrustes alignment



To calculate

```
>  consensus <- apply(proc$coords, c(1,2), mean)

> plot(consensus,asp=1, type="n")

> for(i in 1:length(proc$coords[,,3]))
points(proc$coords[,,i])

> points(consensus, col=”Red”, cex=2, pch=20)
```

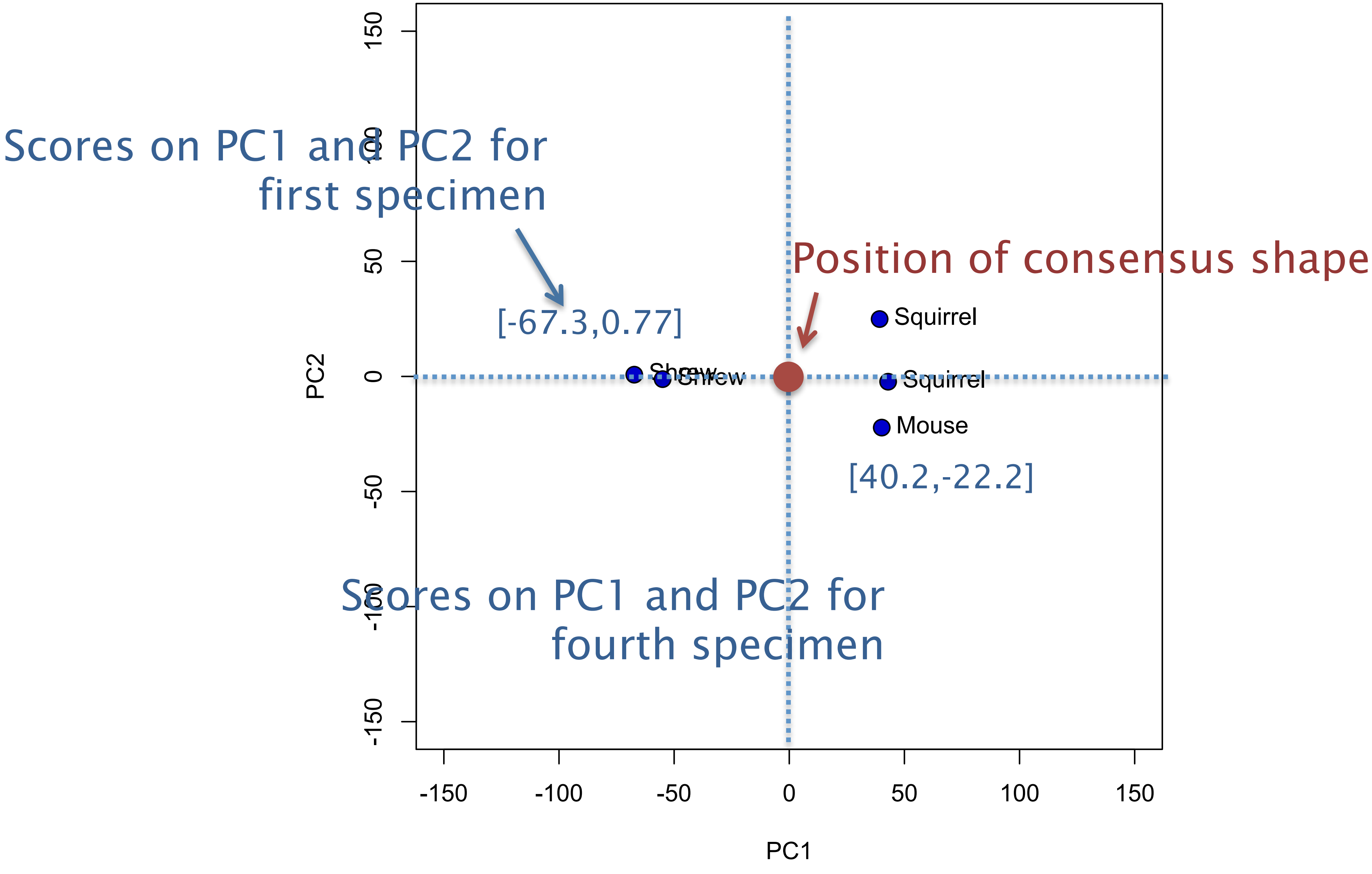
Terminology: centroid

The mean (center) of a shape (or of a landmark).
Average x,y of all coordinates across all landmarks.
(for individual landmark, the centroid is the consensus point for that landmark).

To calculate:

```
> centroid <- apply(proc$coords,2,mean)
```

Finding your way around shape space



Scores on PC1 and PC2 for first specimen

[-67.3,0.77]

Position of consensus shape

[40.2,-22.2]

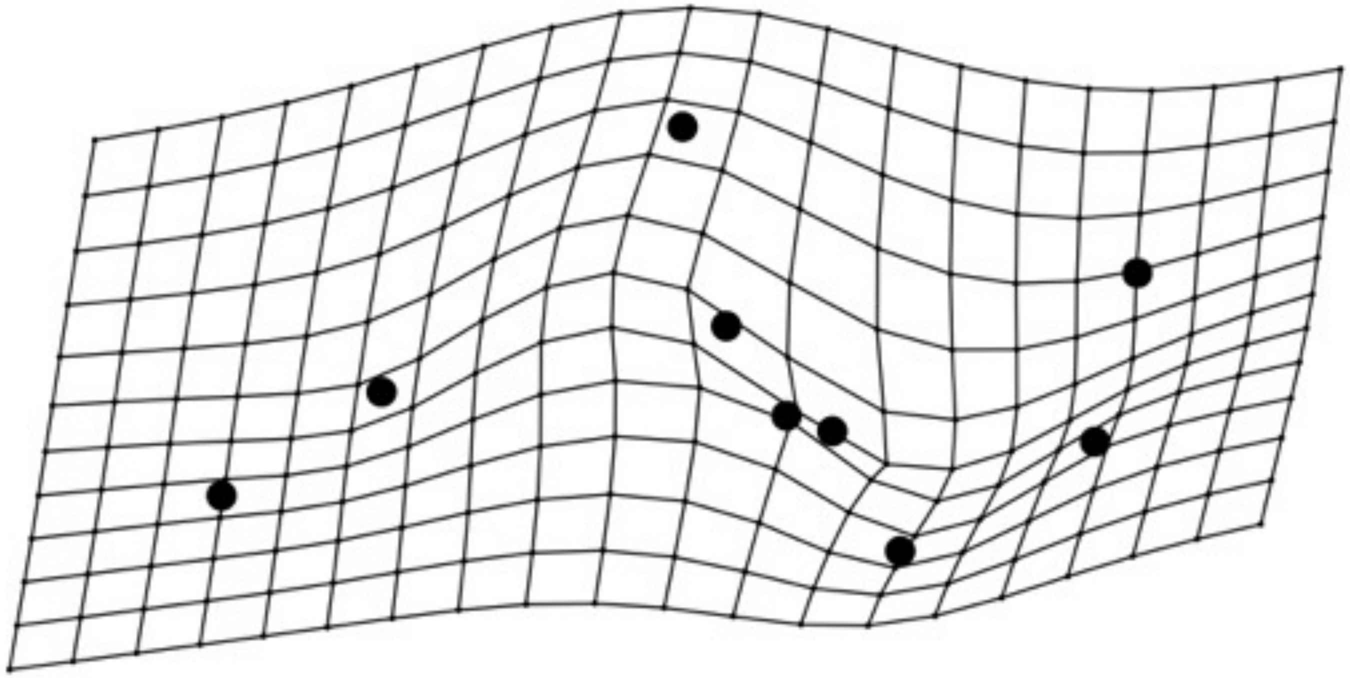
Scores on PC1 and PC2 for fourth specimen

Visualizing shapes: thin plate spline grids

Allows difference between two shapes to be compared as a grid of first being deformed into second.

Usually differences are shown between real (or imaginary) specimens and the consensus shape.

```
> plotRefToTarget(consensus,proc
$coords[,,1])
```



Principles of shape space

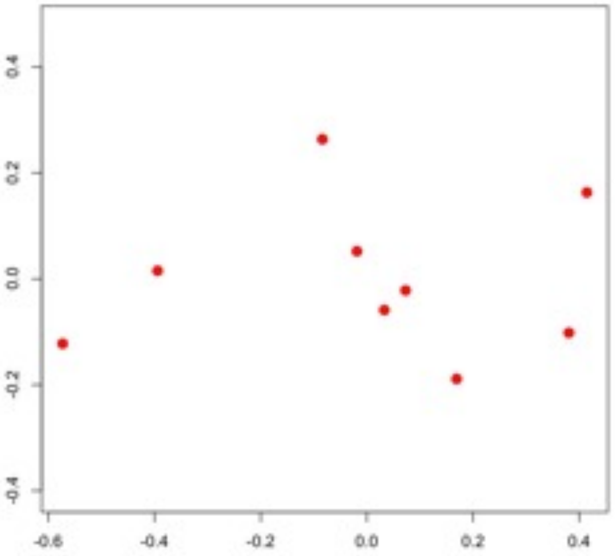
1. PCA space = shape space
2. Every point in shape space corresponds to configuration of landmarks
3. By definition, the consensus (mean shape) lies at the origin (0,0) of the shape space
4. PC scores are the coordinates of the PCA plot (they are the “addresses” of the points in the shape space)

Landmark Space (specimen space)

Two (or three) dimensional space (x, y, z)

Numbers associated with space are the landmark coordinates

Many points per specimen

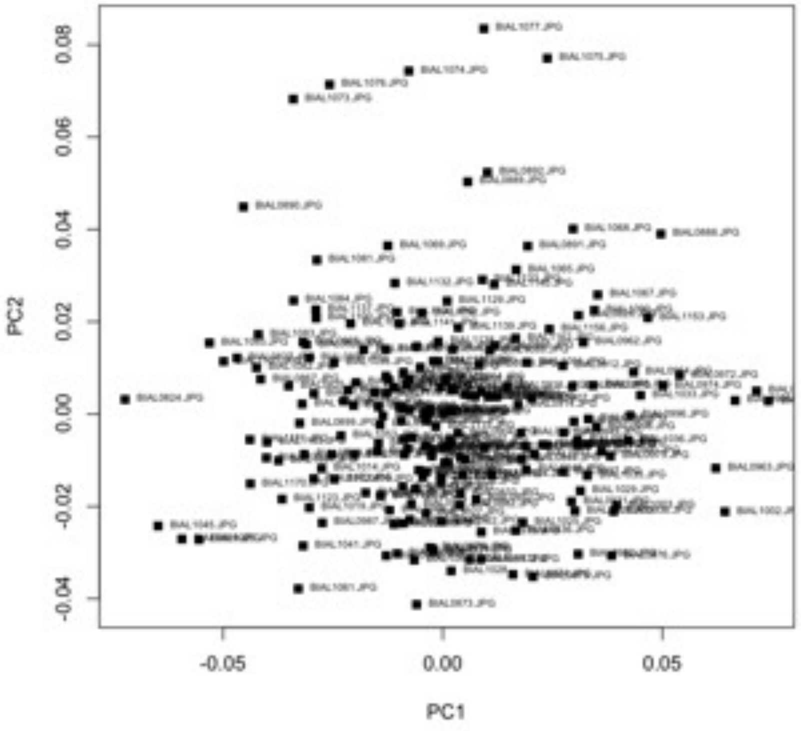


Shape Space (PCA space)

Multidimensional space (nearly as many dimensions as there are landmarks x coordinates)

Numbers are scores (addresses in PC space)

One point per specimen



Basic steps of shape analysis in *R*

Capture images
digital camera, etc.

Collect coordinates
digitize2d(*filename*)

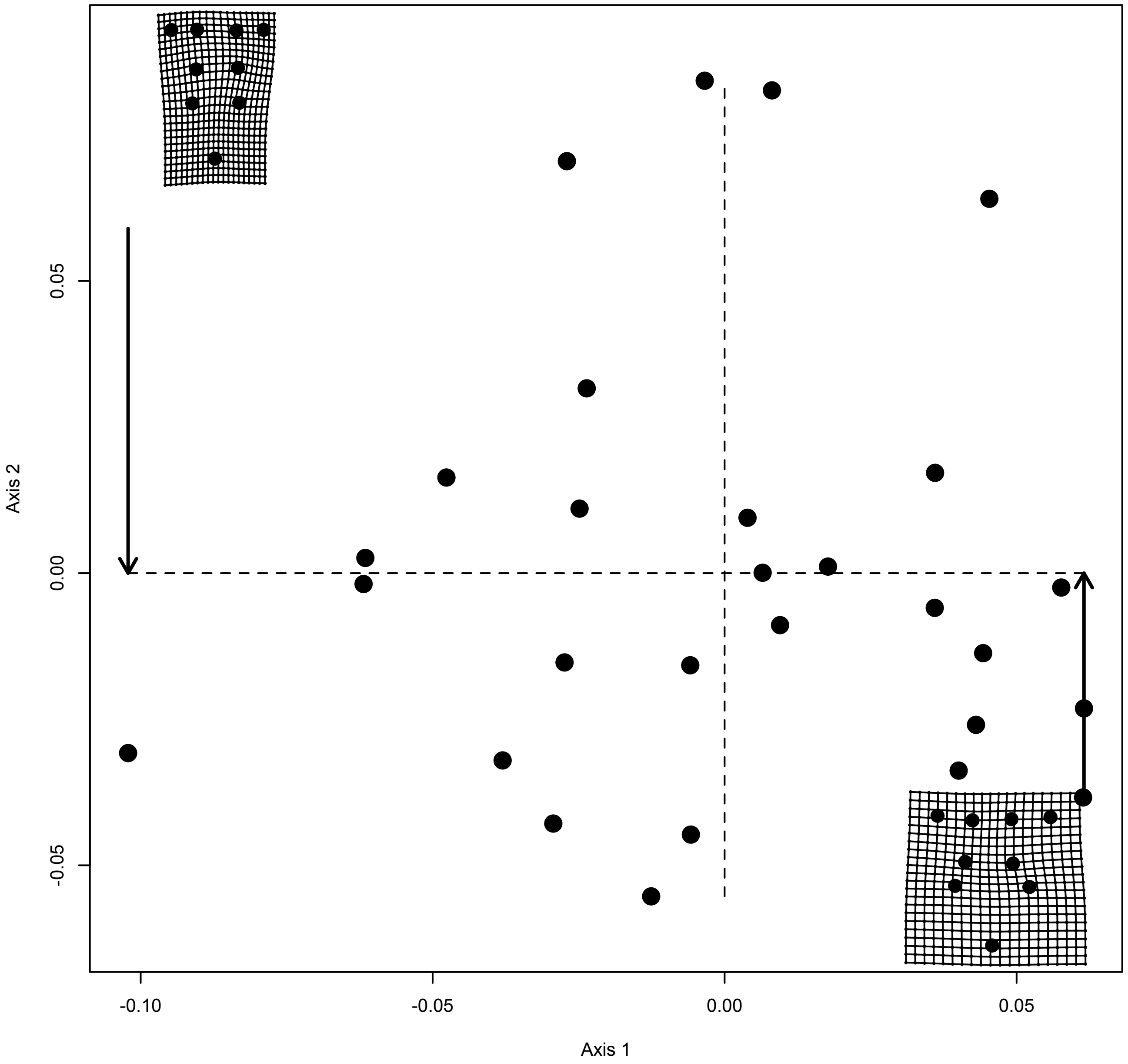
Read coordinate files in NTS format
readmulti.nts(*filenames*)

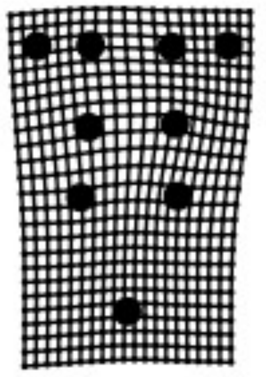
Superimpose landmarks and do PCA
gpagen(*lands*)

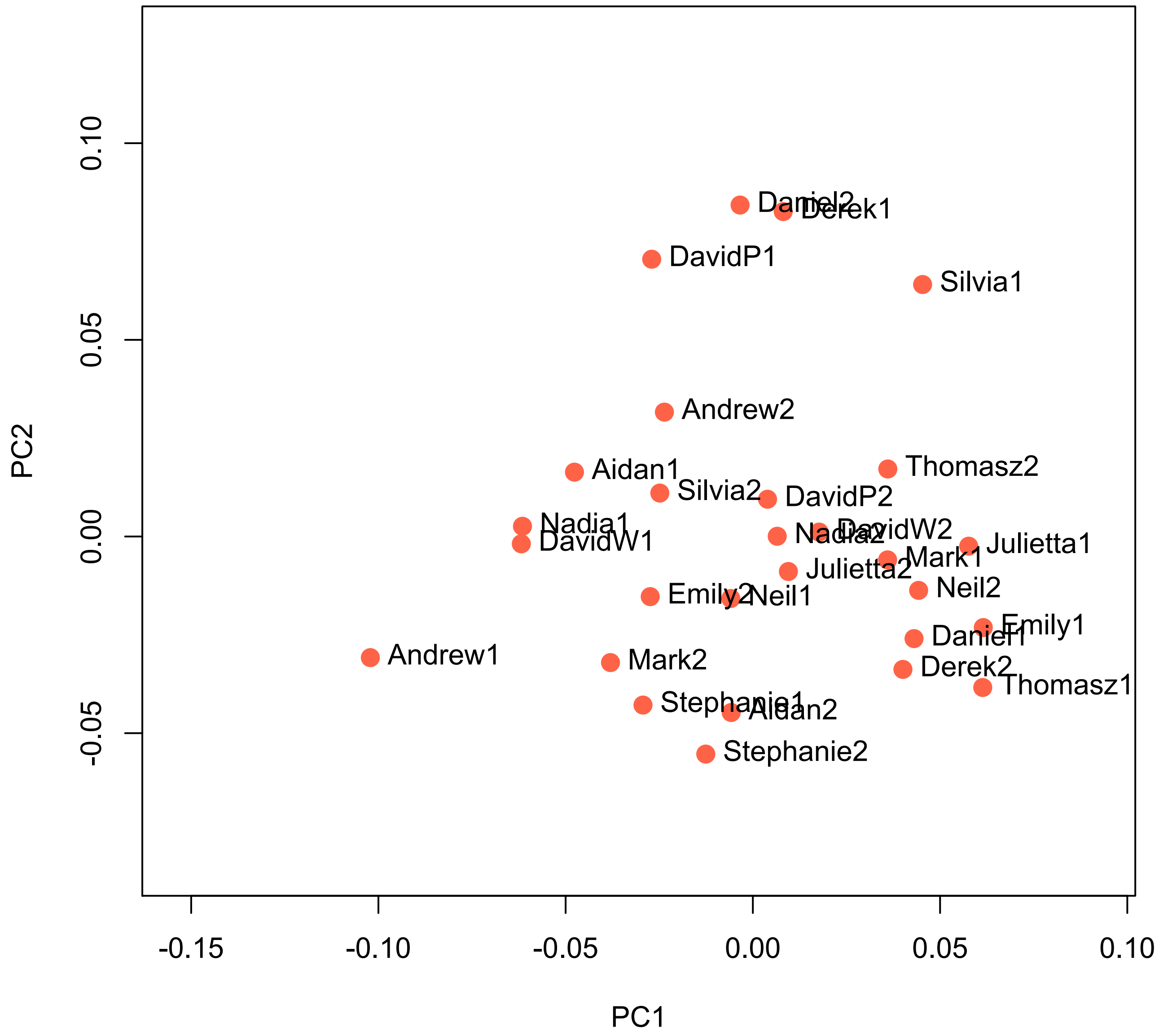
View PCA results
plotTangentSpace(*$coords*)

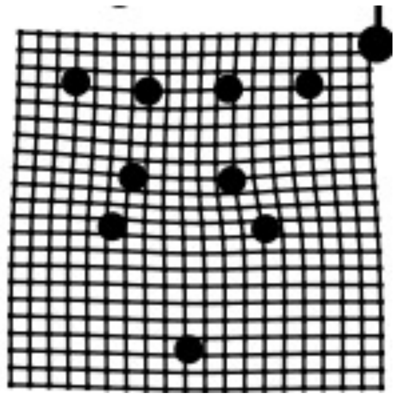
Look at shape differences
plotRefToTarget(*shape1, shape2*)

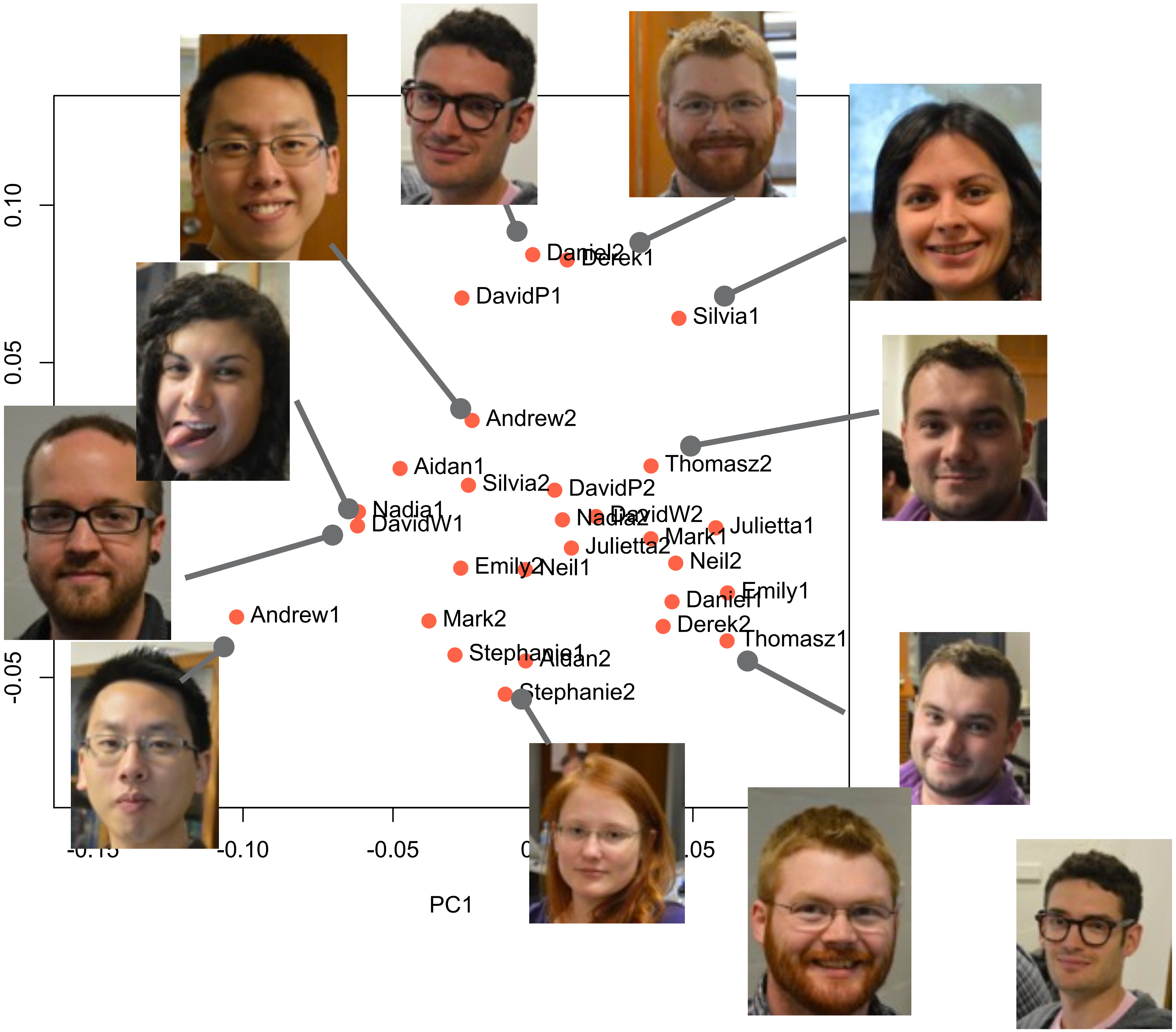
PCA of faces






Daniel2
Derek1
DavidP1
Silvia1
Andrew2
Aidan1
Thomasz2
Silvia2
DavidP2
Nadia1
DavidW1
Nadia2
DavidW2
Julietta1
Mark1
Julietta2
Neil2
Emily2
Neil1
Emily1
Daniel1
Andrew1
Mark2
Derek2
Thomasz1
Stephanie1
Aidan2
Stephanie2
0.10
0.05
0.00
-0.05
PC2
-0.15
-0.10
-0.05
0.00
0.05
0.10
PC1




PC2
0.10
0.05
-0.05
PC1
-0.15
-0.10
-0.05
0
0.05
Daniel2
Derek1
DavidP1
Silvia1
Andrew2
Aidan1
Silvia2
DavidP2
Thomasz2
Nadia1
DavidW1
Nadia2
DavidW2
Julietta1
Mark1
Julietta2
Neil2
Emily2
Neil1
Emily1
Daniel1
Derek2
Andrew1
Mark2
Thomasz1
Stephanie1
Aidan2
Stephanie2