Describing leaf shape using ellipitical forier analysis

Analysis and visualization performed by Ciera Martinez and Daniel Chitwood.

Purpose

The purpose analysis is to investigate how leaf shape differs in the *entire-2* mutant compared to wild type.

```
# Requisite libraries
library(Momocs)
```

```
##
## Attaching package: 'Momocs'
##
## The following object is masked from 'package:utils':
##
##     stack
```

```
library(shapes)
```

```
## Loading required package: scatterplot3d
## Loading required package: rgl
## Loading required package: MASS
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.1.3
```

```
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
##
## The following object is masked from 'package:MASS':
##
##     select
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# Function to convert SHAPE nef file to a Momocs coe object
NEF2COE<-function (nef.path){
  nef <- readLines(nef.path)
  HARMO.l <- grep(pattern = "HARMO", nef)
  nb.h <- as.numeric(substring(nef[HARMO.l], 8))
  nef <- nef[-(1:HARMO.l)]
  nb.coo <- length(nef)/(nb.h + 1)
  coo.i <- 1:nb.coo
  coo.beg <- (coo.i - 1) * (nb.h + 1) + 1
  coo.end <- coo.beg + nb.h
  res <- matrix(NA, nrow = nb.coo, ncol = nb.h * 4,
      dimnames = list(nef[coo.beg], paste(rep(LETTERS[1:4], each = nb.h),
      1:nb.h, sep = "")))
  for (i in seq(along = coo.i)) {
    nef.i <- nef[(coo.beg[i]+1):coo.end[i]]
    x <- as.numeric(unlist(strsplit(nef.i, " ")))
    x1<-x[!is.na(x)]
    a.i<-x1[seq(1,length(x1),4)]
    b.i<-x1[seq(2,length(x1),4)]
    c.i<-x1[seq(3,length(x1),4)]
    d.i<-x1[seq(4,length(x1),4)]
    res[i, ]<-c(a.i,b.i,c.i,d.i)
  }
  return(Coe(res,method="eFourier"))}
```

Load in Data

```
#Create the Momocs Coe object from the SHAPE NEF file
e_coe <- NEF2COE("../data/e2_wt_NEF.txt")

#Retrieve the names associated with the harmonics of each leaflet,
#so that we can retrieve the factor levels (entire vs. wild type)
#so that we can compare the two genotypes
names <- e_coe@names
str(names)
```

```
##  chr [1:1430] "L1_EA1_1" "L1_EA10_1" "L1_EA11_1" "L1_EA12_1" ...
```

I can't look up how to do this in R at the moment but I need to extract the genotype and leaf number information later

```
# Extract genotypes from names
namesTable <- as.data.frame(names)
colnames(namesTable)[1] <- "id"

#set genotype
namesTable$genotype <- ifelse(grepl("_E", namesTable$id, ignore.case = T), "e2",
        ifelse(grepl("R", namesTable$id, ignore.case = T), "wt", "wt"))

# Get leaf column
namesTable <- separate(namesTable, id, into = c("leaf", "id"), sep = 2)

#remove id
```

```
namesTable <- namesTable[,-2]

##Make new id column
namesTable$id <- unite(namesTable, "id", leaf, genotype, sep = "_")
```

Read back in the genotypic factor levels. E = entire, W = wild type, and L = leaf number, indicated by the following number. Row order matters.

```
genotype <- read.table("../data/genotypes.txt", header=TRUE)

#Add the genotype factor levels to the coe object
e_coe@fac <- genotype

#Check that the factor levels have been added to the coe object @fac slot
str(e_coe)
```

```
## Formal class 'Coe' [package "Momocs"] with 5 slots
##   ..@ coe   : num [1:1430, 1:80] 0.849 0.855 0.878 0.94 1.046 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:1430] "L1_EA1_1" "L1_EA10_1" "L1_EA11_1" "L1_EA12_1" ...
##   .. .. ..$ : chr [1:80] "A1" "A2" "A3" "A4" ...
##   ..@ names : chr [1:1430] "L1_EA1_1" "L1_EA10_1" "L1_EA11_1" "L1_EA12_1" ...
##   ..@ fac   :'data.frame':   1430 obs. of  1 variable:
##   .. ..$ genotype: Factor w/ 8 levels "L1_E","L1_W",..: 1 1 1 1 1 1 1 1 1 1 ...
##   ..@ nb.h  : num 20
##   ..@ method: chr "eFourier"
```

Morphometrics

PCA

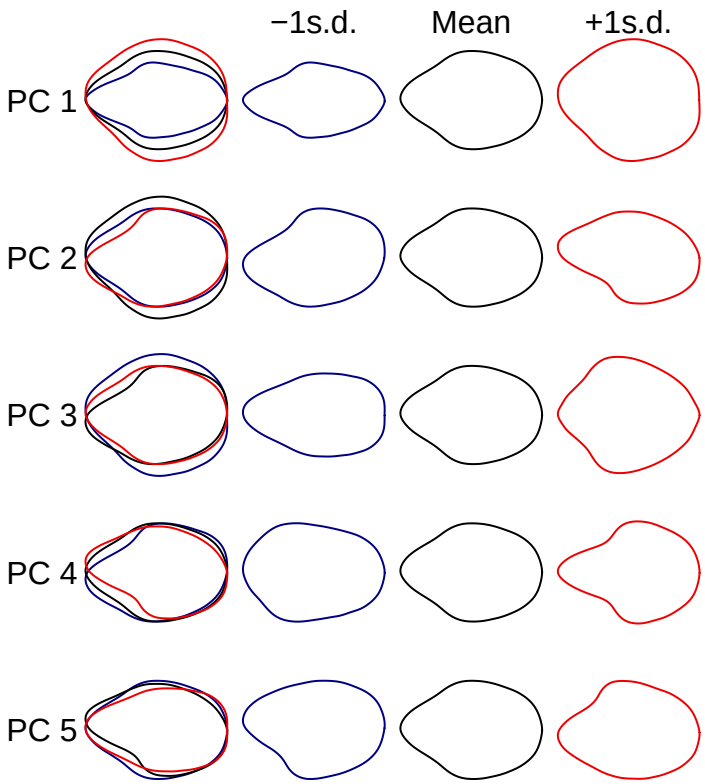
```
#Let us first do a PCA to get a feel for the shape variance, and how it's modulated by genotype
epca <- pca(e_coe)
```

What is the percent variance explained by each principal component? Note, these are eigenvalues, so you'll have to sum all of these values, then recalculate each as a percentage of that sum to figure out the percent

```
#variance explained by each principal component
eigenvalues <- epca$eig
```

Let's check out the eigenleaves. Unlike in SHAPE, you have more control over the visualization of eigenleaves, and can specify how many standard deviations you want to visualize leaves at for each principal component.

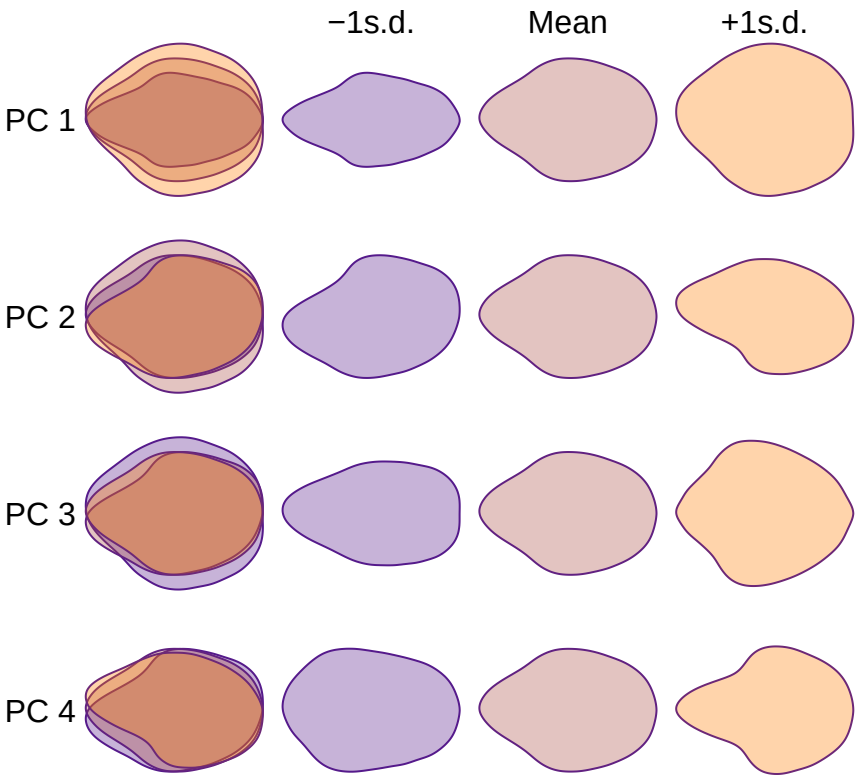
```
#1 standard deviation seems to be adequate for this dataset
PC.contrib(epca, PC.r=1:5, sd=1)
```



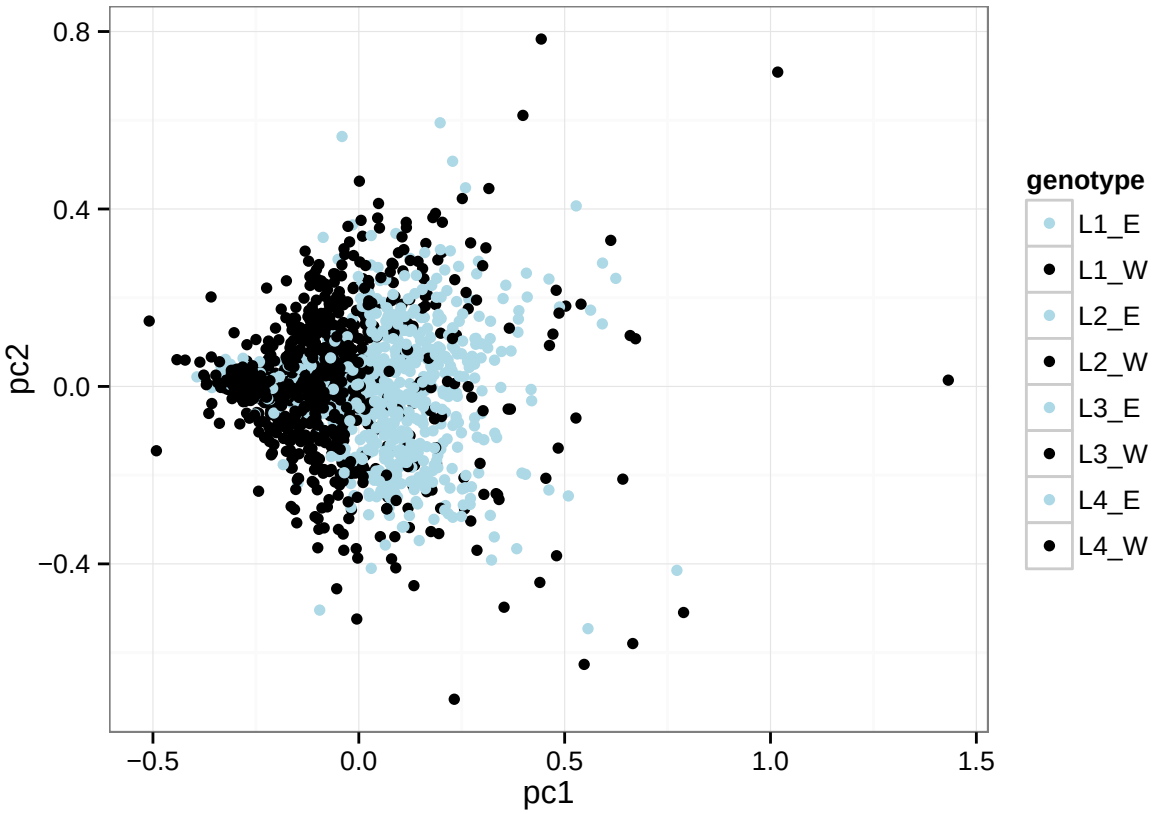

```
args(PC.contrib)
```

```
## function (dudi, PC.r = 1:dudi$nf, sd = 2, cols = rep(NA, 3),
##     borders = c("#000080", "#000000", "#EE0000"), lwd = 1, nb.pts = 300,
##     plot = TRUE, legend = TRUE)
## NULL
```

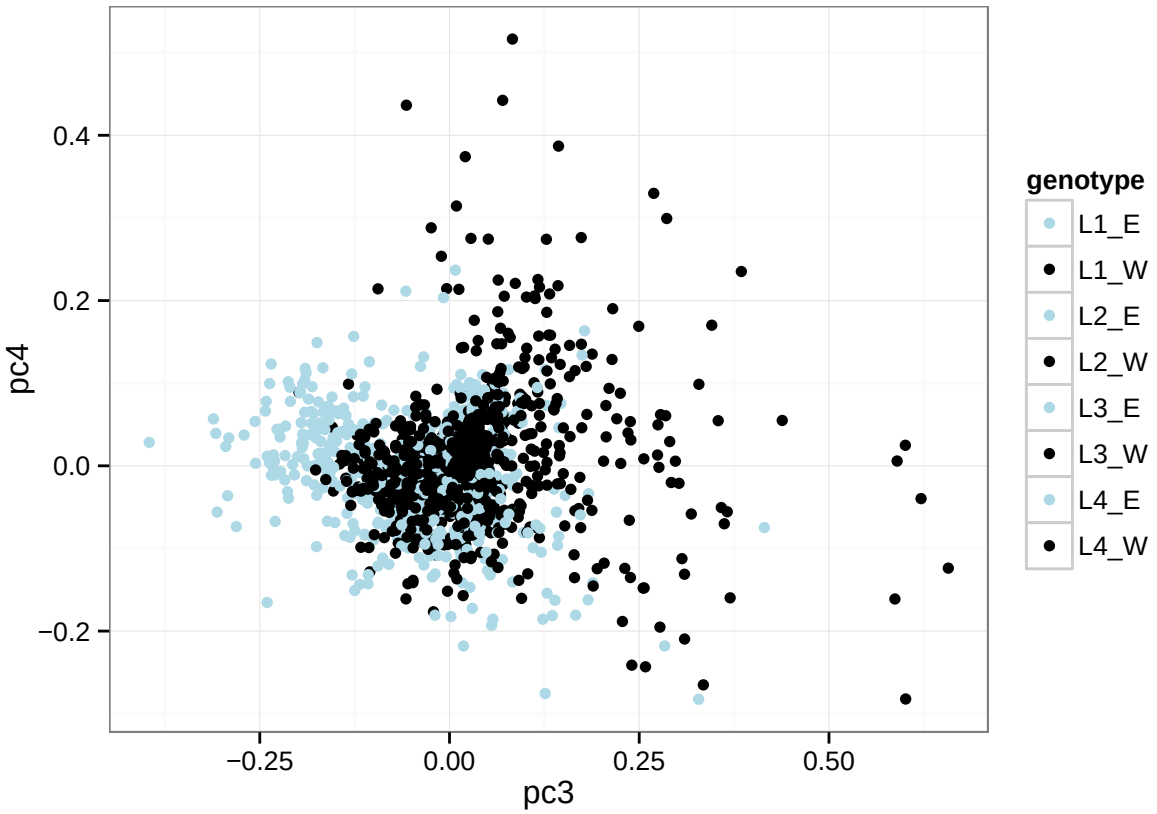
```
PC.contrib(epca, PC.r=1:4, sd=1,
  cols=paste0(col.sari(3), "55"), borders=col.sari(15))
```



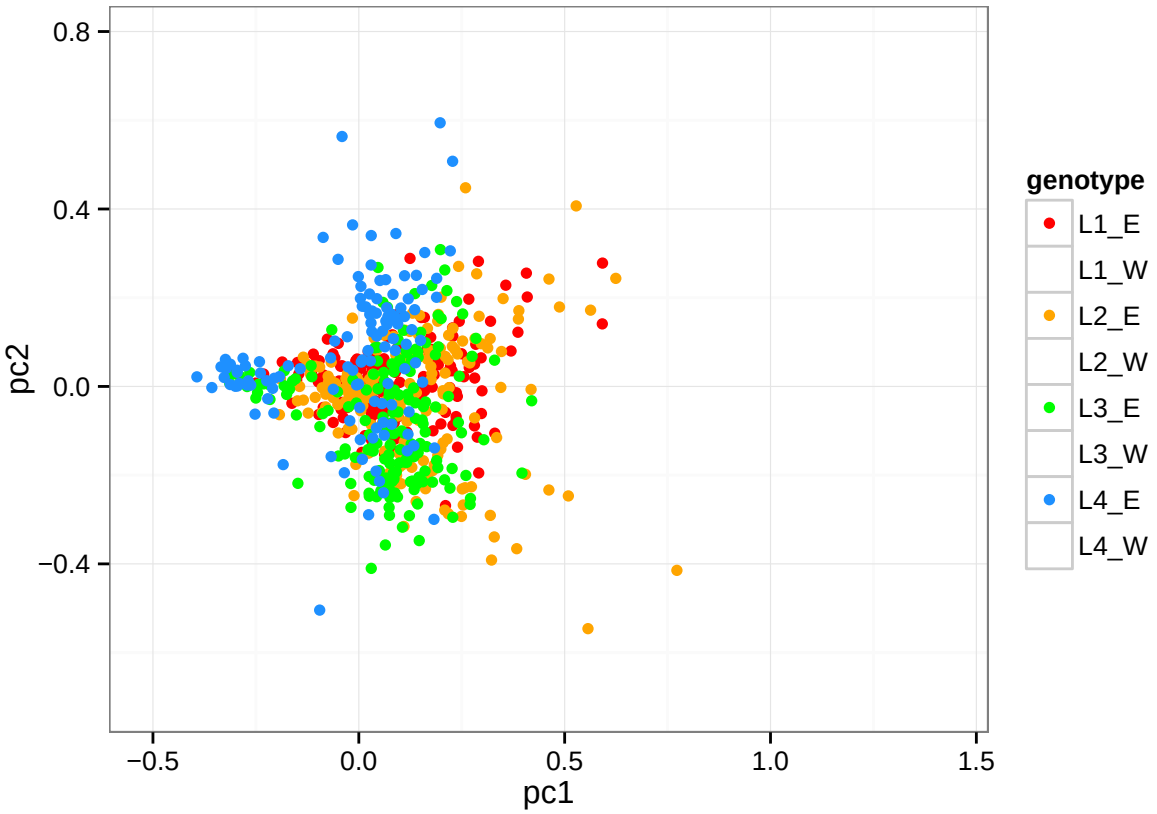

```
#Finally, let's get some PC scores
scores <- epca$li

#Let's get the factor levels back together with the principal component scores to finally do some visua
epca_fac <- cbind(scores, genotype)

#I want to rename the columns
colnames(epca_fac) <- c("pc1", "pc2", "pc3", "pc4", "pc5", "genotype")

#By genotype. PC1, PC2. WOW! Great separation of entire vs. WT by PC1!!!
p <- ggplot(epca_fac, aes(pc1, pc2, colour=genotype))
p + geom_point() + scale_colour_manual(values=c("light blue","black","light blue","black","light blue",
```



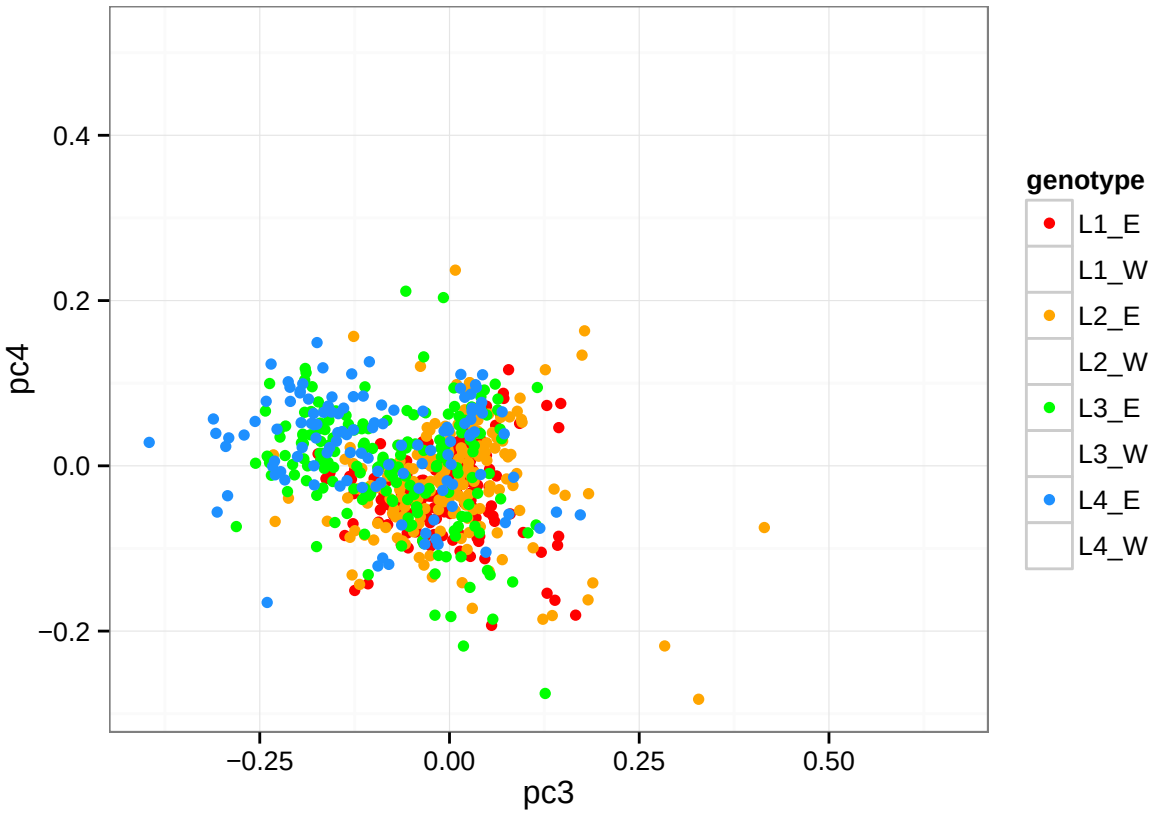

```
#By genotype. PC3, PC4
p <- ggplot(epca_fac, aes(pc3, pc4, colour=genotype))
p + geom_point() + scale_colour_manual(values=c("light blue","black","light blue","black","light blue",
```



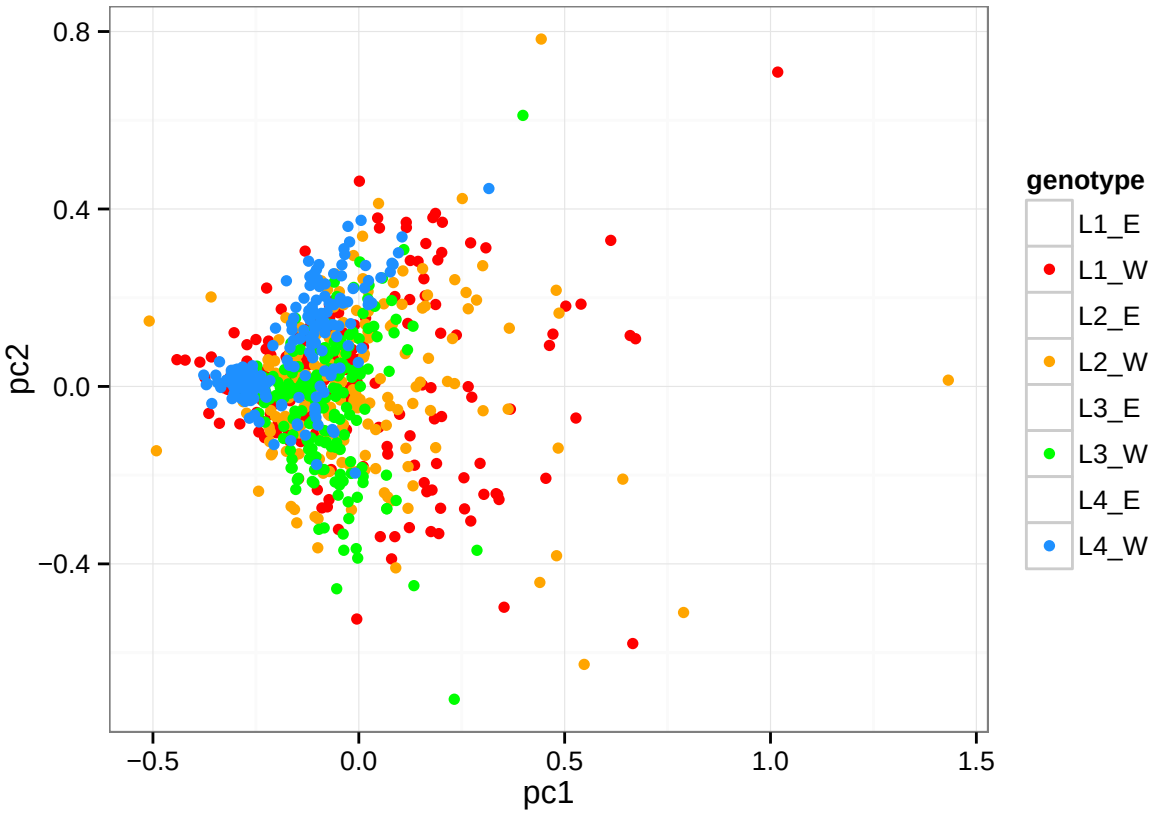

```
#Only entire, by leaf number. PC1, PC2
p <- ggplot(epca_fac, aes(pc1, pc2, colour=genotype))
p + geom_point() + scale_colour_manual(values=c("red","NA","orange","NA","green","NA","dodgerblue","NA"
```



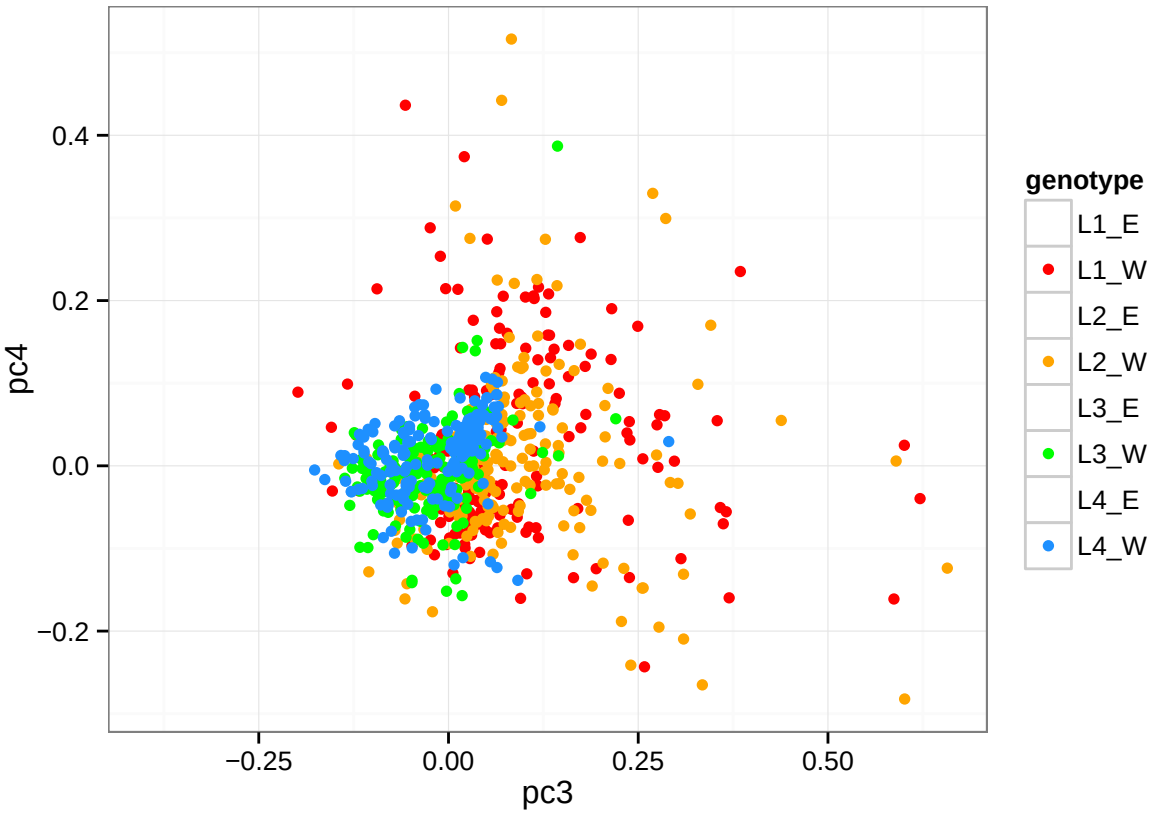

```
#Only entire, by leaf number. PC3, PC4
p <- ggplot(epca_fac, aes(pc3, pc4, colour=genotype))
p + geom_point() + scale_colour_manual(values=c("red","NA","orange","NA","green","NA","dodgerblue","NA"
```




```
#Only wild type, by leaf number. PC1, PC2
p <- ggplot(epca_fac, aes(pc1, pc2, colour=genotype))
p + geom_point() + scale_colour_manual(values=c("NA","red","NA","orange","NA","green","NA","dodgerblue"
```




```
#Only wild type, by leaf number. PC3, PC4
p <- ggplot(epca_fac, aes(pc3, pc4, colour=genotype))
p + geom_point() + scale_colour_manual(values=c("NA","red","NA","orange","NA","green","NA","dodgerblue"
```

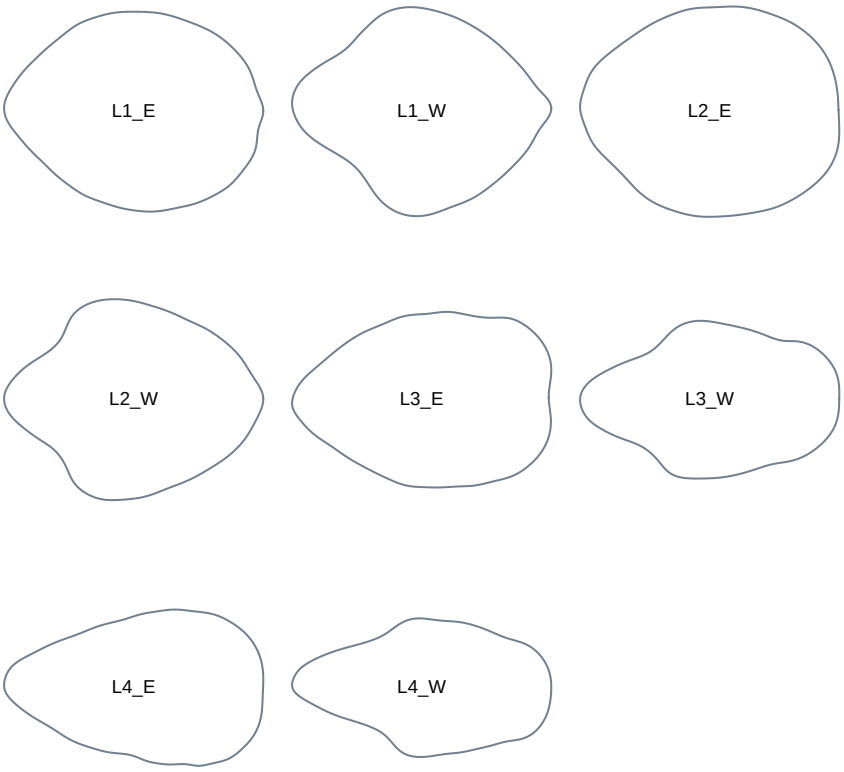


Splines

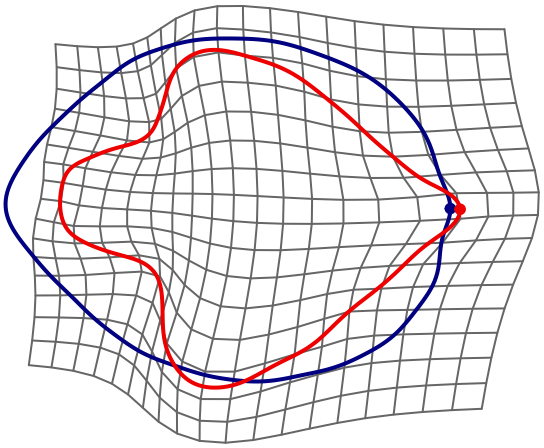
But each PC only captures a fraction of the shape variance. Let us visualize the totality of shape differences between entire and wild type using thin plate splines, inspired by d'Arcy.

We begin by taking the mean shape of each factor level. This is as simple as averaging the harmonic coefficients for each levels. Note: At the moment, the factor levels are for each leaf for each genotype. If you want to compare genotypes by all leaves, then you will have to refactor the levels to be entire and wild type, regardless of leaf

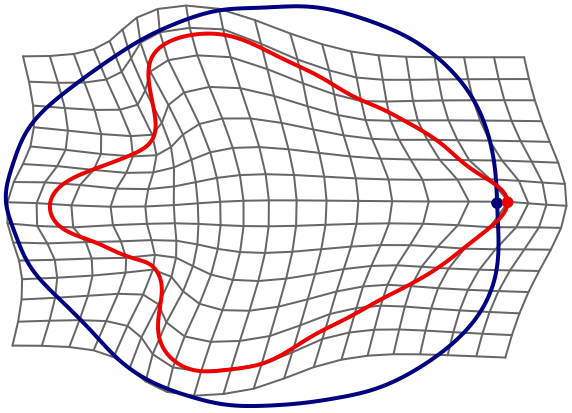
```
mean_leaves <- meanShapes(e_coe)

#Let's first just look at the mean shapes for each genotype, by each leaf
panel(Coo(mean_leaves), names=TRUE)
```



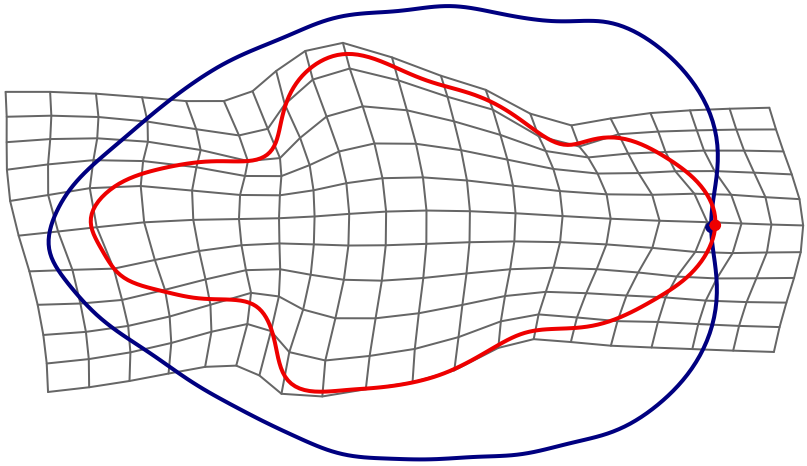

```
#Now, let's make a thin plate spline comparing each genotype
#against the other, for each leaf. Note, you can amplify the
#differences between the mean shapes with the "amp" argument.
#We don't need amplification, so let's keep it at 1

#Thin plate spline, entire vs. wild type, leaf 1
tps.grid(mean_leaves$L1_E, mean_leaves$L1_W, amp=1)
```



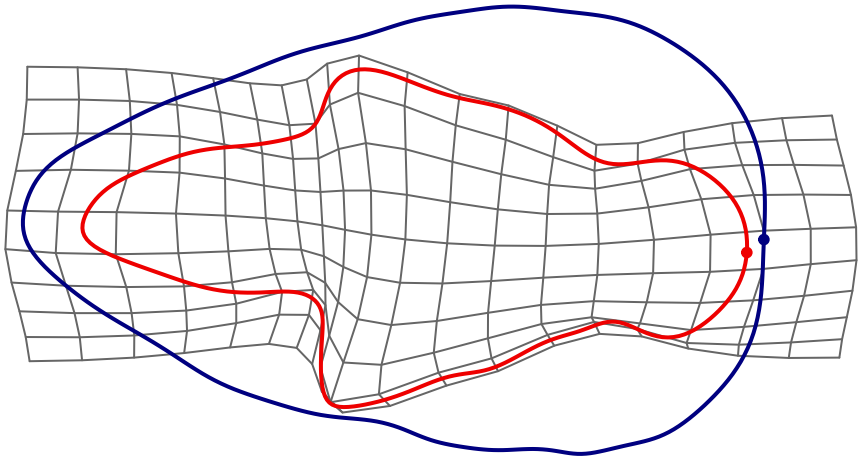
```
#Thin plate spline, entire vs. wild type, leaf 2
tps.grid(mean_leaves$L2_E, mean_leaves$L2_W, amp=1)
```



```
#Thin plate spline, entire vs. wild type, leaf 3
tps.grid(mean_leaves$L3_E, mean_leaves$L3_W, amp=1)
```



```
#Thin plate spline, entire vs. wild type, leaf 4
tps.grid(mean_leaves$L4_E, mean_leaves$L4_W, amp=1)
```



Linear Discrimination Analysis (LDA)

What we really want to do instead of a PCA is maximize the separation of leaflet shapes, using their harmonic coefficients, as a function of genotype. This will use all available shape information to maximize separation by genotype, finding those shape attributes most responsible for discriminating by genotype. To do this, we use a linear discriminant analysis (LDA).

I want to discriminate by genotype only, regardless of which leaf a leaflet comes from. For that, I need different factor levels again. The genotype only information (no leaf info) is in "genotype_only.txt"

```
geno_only <- read.table("../data/genotype_only.txt", header=TRUE)

#Let's put the harmonic coefficents and the genotype only factor levels together
h <- as.data.frame(e_coe@coe)
x <- cbind(h,geno_only)
names(x)
```

```
##  [1] "A1"       "A2"       "A3"       "A4"       "A5"       "A6"
##  [7] "A7"       "A8"       "A9"       "A10"      "A11"      "A12"
## [13] "A13"      "A14"      "A15"      "A16"      "A17"      "A18"
## [19] "A19"      "A20"      "B1"       "B2"       "B3"       "B4"
## [25] "B5"       "B6"       "B7"       "B8"       "B9"       "B10"
## [31] "B11"      "B12"      "B13"      "B14"      "B15"      "B16"
## [37] "B17"      "B18"      "B19"      "B20"      "C1"       "C2"
## [43] "C3"       "C4"       "C5"       "C6"       "C7"       "C8"
## [49] "C9"       "C10"      "C11"      "C12"      "C13"      "C14"
## [55] "C15"      "C16"      "C17"      "C18"      "C19"      "C20"
## [61] "D1"       "D2"       "D3"       "D4"       "D5"       "D6"
## [67] "D7"       "D8"       "D9"       "D10"      "D11"      "D12"
## [73] "D13"      "D14"      "D15"      "D16"      "D17"      "D18"
## [79] "D19"      "D20"      "genotype"
```

```
#Perform the LDA
lda <- lda(genotype~A1+A2+A3+A4+A5+A6+A7+A8+A9+A10+A11+A12+A13+A14+A15+A16+A17+A18+A19+A20+B1+B2+B3+B4+
```

Now, using the linear discriminant rules, let's try to reassign leaflets to genotypic classes based on the rules we just created. If there is confusion, it indicates that leaflets can't be discrimnated by genotype. If most leaflets are correctly reassigned, it indicates unique shape attributes separate the two genotypic classes

```
t(table(predict(lda, type="class")$class, x$genotype))
```

```
##
##       E   W
##   E 570 102
##   W  92 666
```

The way you should read this table is: of 672 entire leaflets, 570 (85%) were correctly reassigned as entire and 102 (15%) incorrectly reassigned as WT. Of 758 WT leaflets, 666 (88%) were correctly reassigned as WT and 92 (12%) were incorrectly reassigned as entire. I suspect the results would vary if you did four different lda's by genotype, one for leaflets from each leaf. I bet the reassignment would be the best in leaf 4 and the worst in leaf 1, just a guess. But certainly, entire leaflets are shaped differently than WT

Finally, let's get the linear discriminant scores back and visualize the separation by genotype.

```
lda_scores <- predict(lda, x[1:80])

#Put the scores and factor levels together
df_lda_scores <- as.data.frame(lda_scores$x)
visual_lda <- cbind(df_lda_scores, geno_only)

#Visualize LDA spearation in ggplot2

p <- ggplot(visual_lda, aes(x=LD1, fill=genotype))
p + geom_density(alpha=0.5) + theme_bw()
```