Introduction to HaugShape Package

Colin Hassenbach

2024-09-30

Introduction

The `HaugShape` package is designed to facilitate advanced shape analysis and visualization. This document demonstrates how to install and use the package to generate plots with features such as convex hulls, heatmaps, and contours.

Installation

To install `HaugShape` from GitHub, run the following command in your R console:

```r
# Install devtools if you don't have it installed
install.packages("devtools")

# Install HaugShape from GitHub
devtools::install_github("TestoKlaus/HaugShape")
```

You can ignore or run the updates for the used packages (But I have not yet tested for all the updated packages).

To load the package, run:

```r
library(HaugShape)
```

Basic Scatterplot

We will use a random data frame like:

```r
# Create a data frame
df <- data.frame(
  PC1 = rnorm(100),
  PC2 = rnorm(100),
  group = sample(c("A", "B", "C"), 100, replace = TRUE)
)

# Check the first few rows of the data frame
head(df)
```

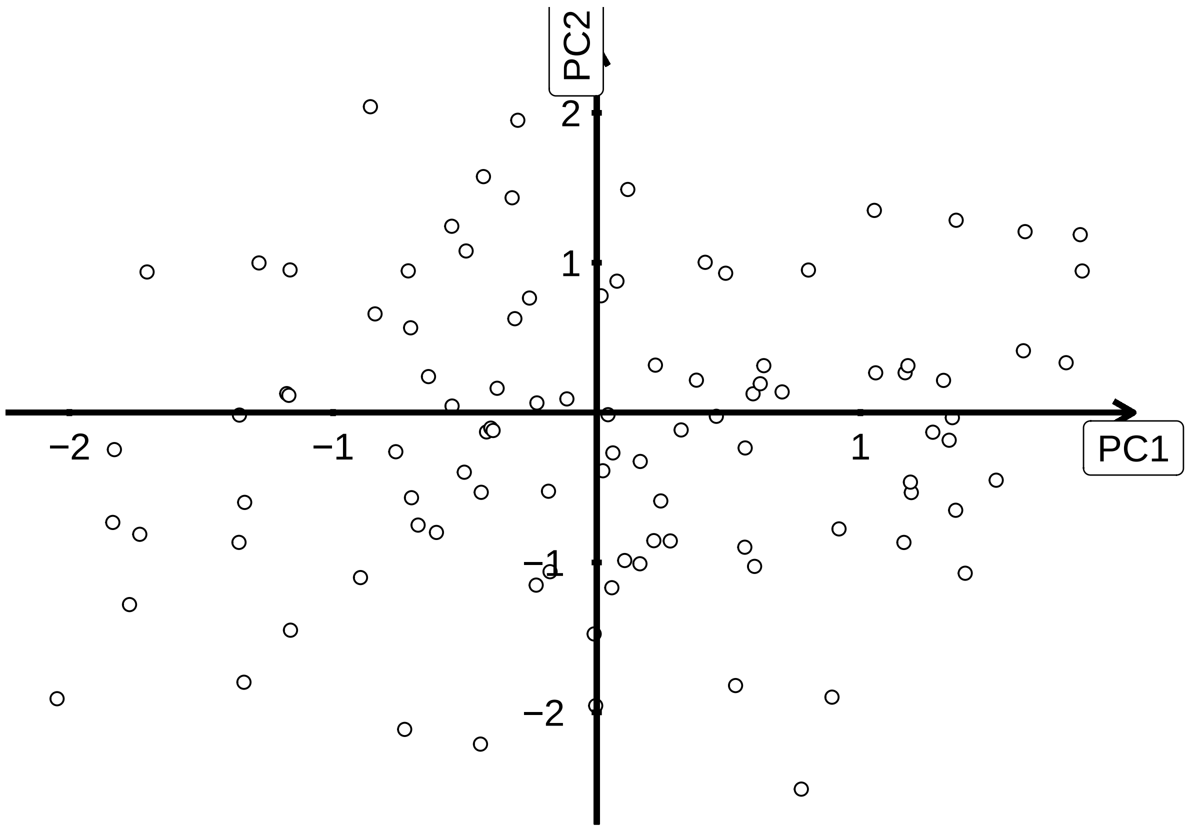
```
##            PC1         PC2 group
## 1  0.04285541 -0.01398823     C
## 2 -1.83544388 -0.73281757     A
## 3 -0.22683749  0.06456688     B
## 4  1.33672769 -0.18427994     B
## 5 -0.32079822  1.43336250     B
## 6  1.51505810 -0.45107929     C
```

This creates a data frame with 3 columns: PC1, PC2, and group. The column 'group' randomly assert the groups A, B, and C in the data frame.

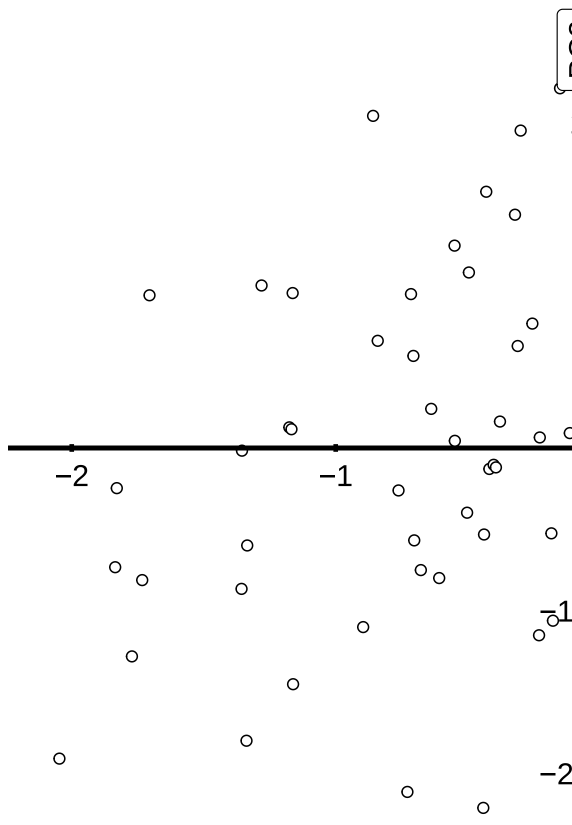
Now let's make a basic scatterplot using the shape_plot() function:

```
shape_plot(data = df, x_col = "PC1", y_col = "PC2", group_col = "group", x_label = "PC1 (...)", y_label
```

Here: `data` = the data frame used, `x_col` = and `y_col` = the columns used for the axes, `group_col` = the column with the groups, `x_label` = and `y_label` = the name of the axis.



The axis labels don't completely fit but we can adjust them using: `x_label_adjust_x` = horizontally shift the x axis label `x_label_adjust_y` = vertically shift the x axis label `x_label_size` = size of the x axis label (Default set to 5) `y_label_adjust_x` = horizontally shift the y axis label `y_label_adjust_y` = vertically shift the y axis label `y_label_size` = size of the y axis label (Default set to 5)



(Or we could simply set a different size when exporting the plot from R)

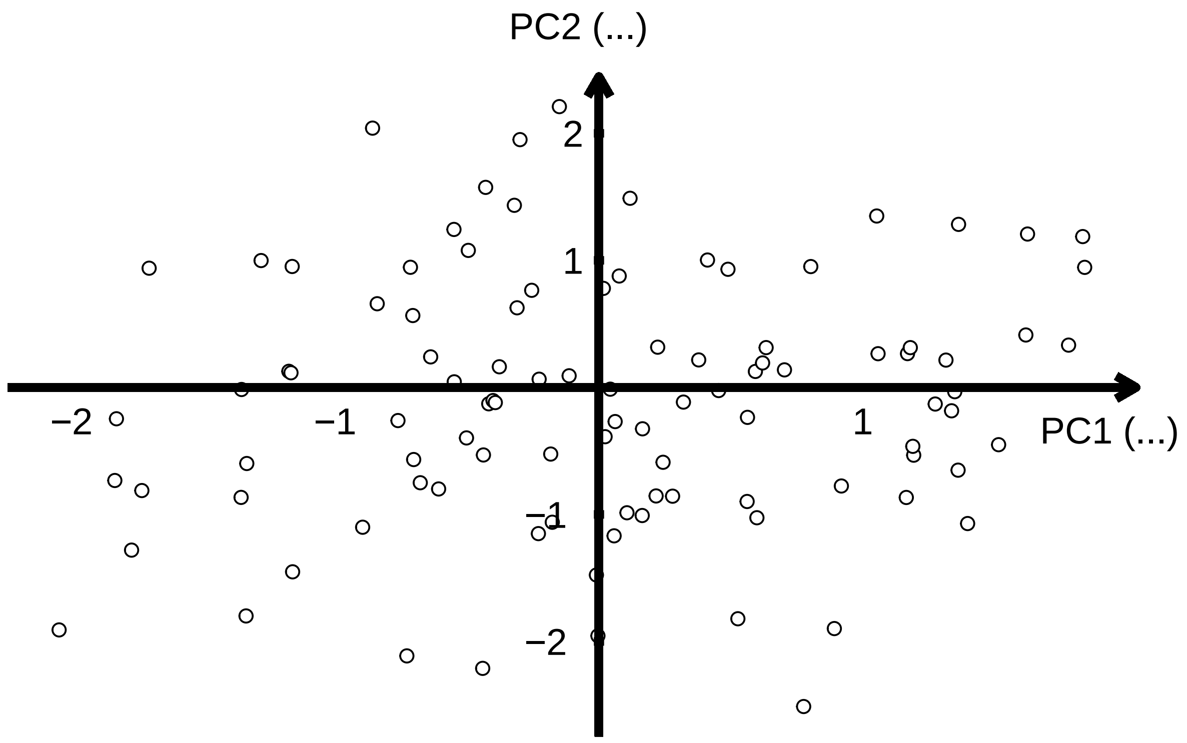
We can also remove the boxes around the labels and rotate the y axis label with: `show_label_text_fields = FALSE` and `rotate_y_label = FALSE`

We can also adjust the line width of the axis with `axis_linewidth =`.

Here the corrected and slightly modified plot (added title with `title = ""`):

```
shape_plot(data = df, x_col = "PC1", y_col = "PC2", group_col = "group",
           x_label = "PC1 (...)", x_label_adjust_x = -0.1, x_label_adjust_y = -0.1,
           y_label = "PC2 (...)", y_label_adjust_y =  0.4, rotate_y_label = FALSE,
           show_label_text_fields = FALSE,
           axis_linewidth = 1.5,
           title = "shape_plot")
```

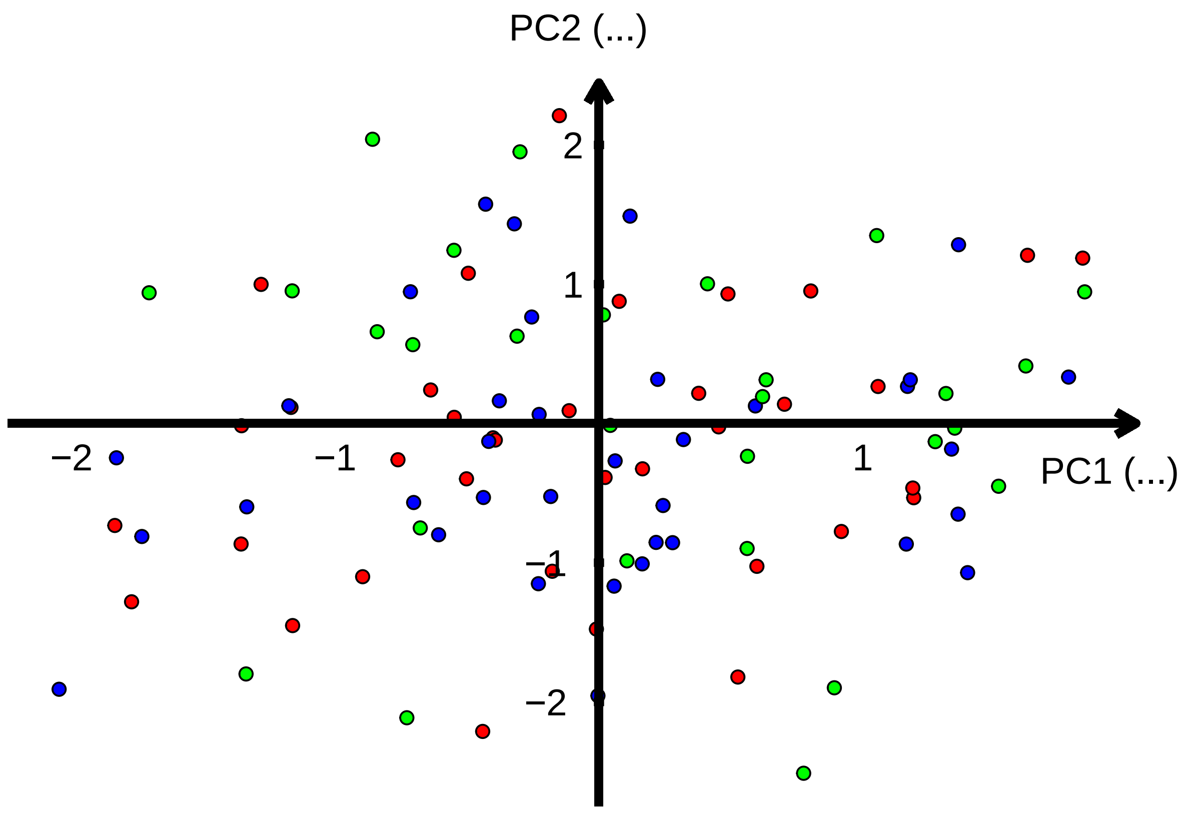
shape_plot



Now that we have nice axis labels, we can continue to edit the plot. For example we can modify the looks of the data points according to group with `point_color =`, `point_fill =`, `point_shape =`, and `point size =`. The values can be either single values or vectors like `point_color = c("red","blue","green")`. `point_color` sets the color of the contour of the point (for `point_shape =` where outline colors can be modified). To adjust the color of the filling you have to use `point_fill = c()`.

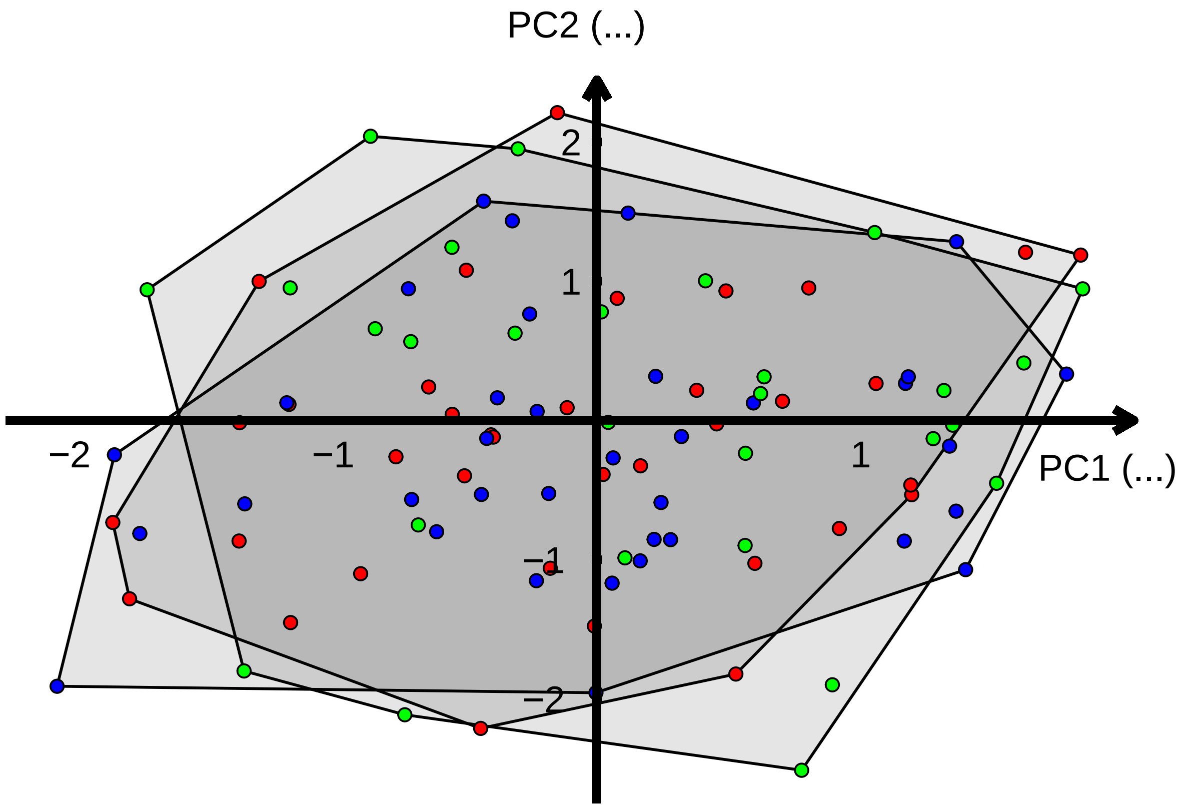
For example, we want to show our 3 groups from the data frame in red, blue, and green. For this, we first have to specify the groups we want to show with `group_vals = c("A","B","C")` (but we could also choose to just show 2 of the groups in the plot with `group_vals = c("A","B")`). Then we have to specify the color with `point_fill = c("red","blue","green")`.

```
shape_plot(data = df, x_col = "PC1", y_col = "PC2", group_col = "group",
           x_label = "PC1 (...)", x_label_adjust_x = -0.1, x_label_adjust_y = -0.1,
           y_label = "PC2 (...)", y_label_adjust_y =  0.4, rotate_y_label = FALSE,
           show_label_text_fields = FALSE,
           axis_linewidth = 1.5,
           group_vals = c("A","B","C"),
           point_fill = c("red","blue","green"))
```



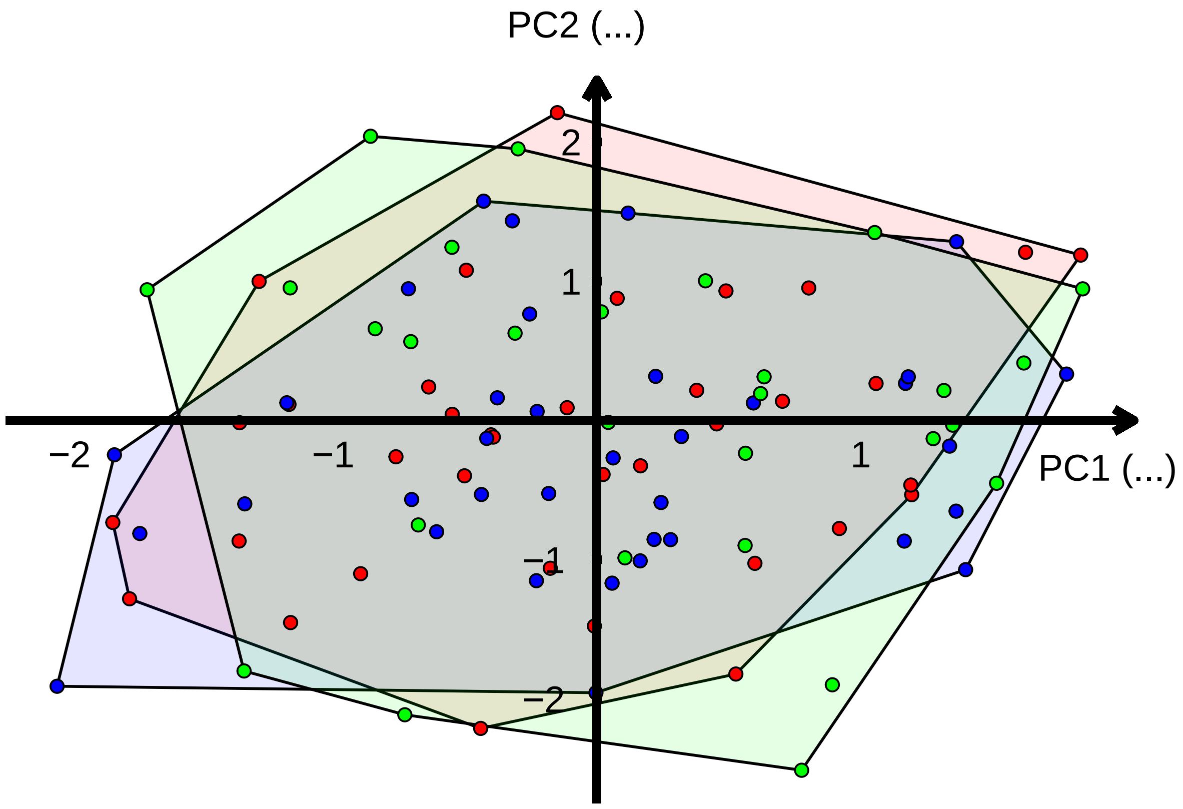
Showing Morpho Spaces

So far, we have just created basic scatterplots, but we often want to show morphospaces. Now that we have our basic scatterplot all set up, we can easily create the morphospaces using `show_hulls = TRUE`



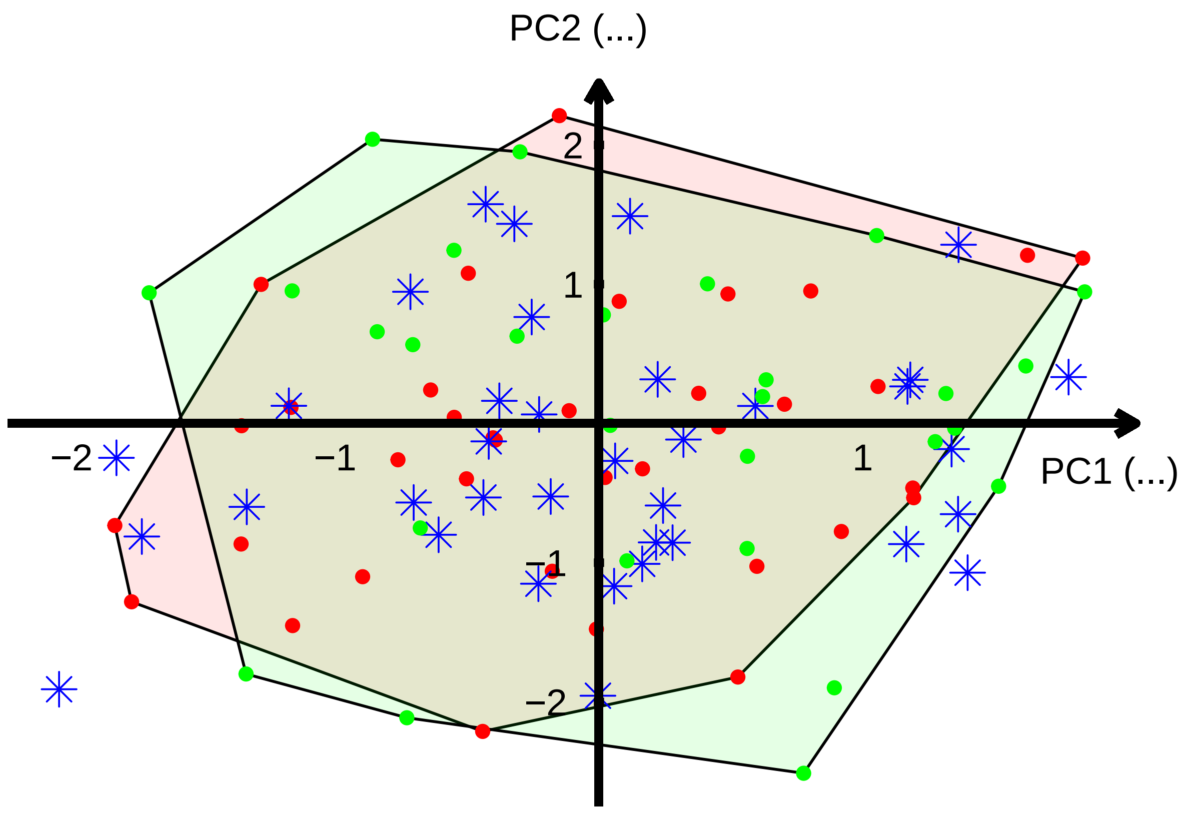
You see that all the hulls look the same. To specify different colors, set a vector of colors for `hull_fill`. You can adjust the outline color of the hull with `hull_color`, the type of the outline with `hull_linetype` (solid is the default), and the transparency with `hull_alpha` (0.1 is the default). I think it would make sense to use the same color as for the points but you can adjust as you like:

```
shape_plot(data = df, x_col = "PC1", y_col = "PC2", group_col = "group",
           x_label = "PC1 (...)", x_label_adjust_x = -0.1, x_label_adjust_y = -0.1,
           y_label = "PC2 (...)", y_label_adjust_y =  0.4, rotate_y_label = FALSE,
           show_label_text_fields = FALSE,
           axis_linewidth = 1.5,
           group_vals = c("A","B","C"),
           point_fill = c("red","blue","green"),
           show_hulls = TRUE,
           hull_fill = c("red","blue","green"))
```



Perhaps we don't want to show the morpho spaces for all groups. In such cases, we can use `show_hull_for_groups = c()` to specify which hulls we want to show exactly. Here, we want to only show the hulls for group A and C. Please note, that I have also changed the point shape (for group B) and the size (for group B). I also set the contour colors for the points to match their filling (the color of `point_shapes` with no contours also can be changed with `point_color`).

```
shape_plot(data = df, x_col = "PC1", y_col = "PC2", group_col = "group",
           x_label = "PC1 (...)", x_label_adjust_x = -0.1, x_label_adjust_y = -0.1,
           y_label = "PC2 (...)", y_label_adjust_y =  0.4, rotate_y_label = FALSE,
           show_label_text_fields = FALSE,
           axis_linewidth = 1.5,
           group_vals = c("A","B","C"),
           point_fill = c("red","blue","green"),
           point_shape = c(21,8,21),
           point_color = c("red","blue","green"),
           show_hulls = TRUE,
           hull_fill = c("red","blue","green"),
           show_hull_for_groups = c("A","C"))
```



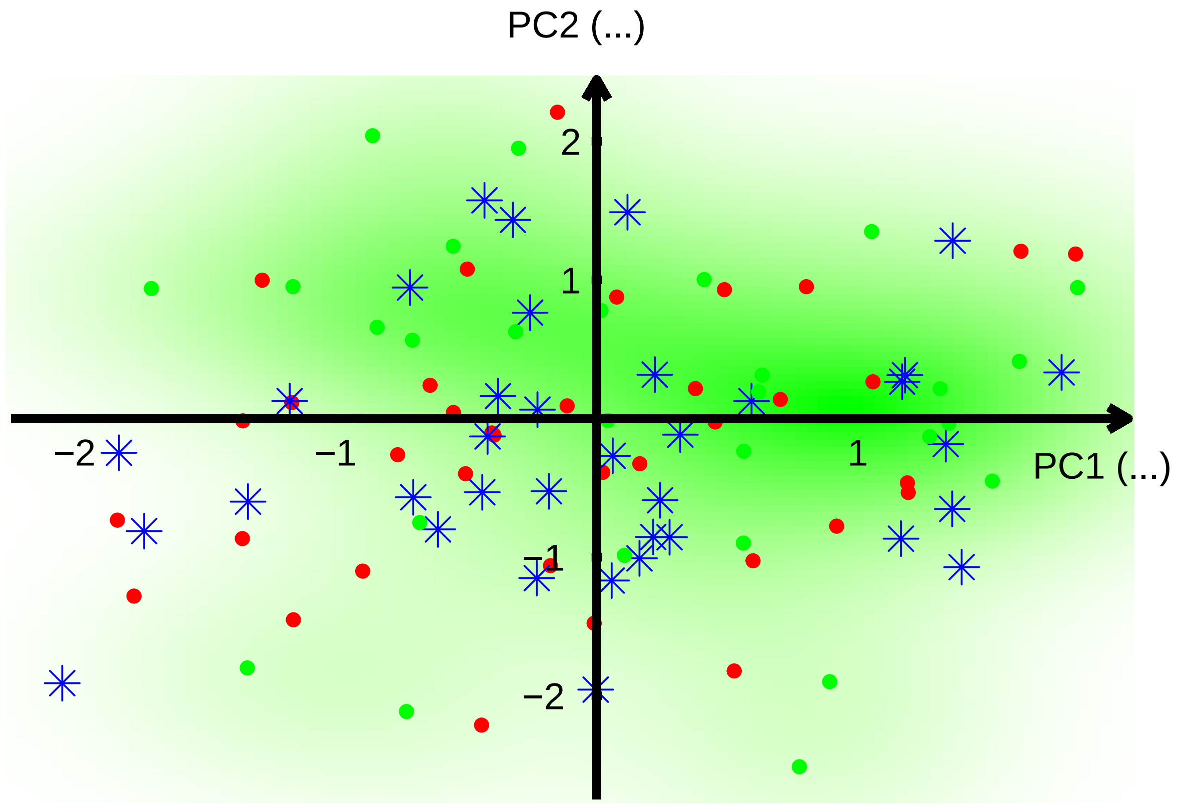
Heatmaps and Density Contours

Heatmaps and density contours follow the same logic as the morpho spaces (hulls).

For Heatmaps: `show_heatmaps =` (whether to show heatmaps or not, default is FALSE), `heatmap_colors =` (colors used in the heatmap, default is `heatmap_colors = list(c("white", "red"), c("white", "blue"))`), and `heatmap_alpha =` (defines the transparency)

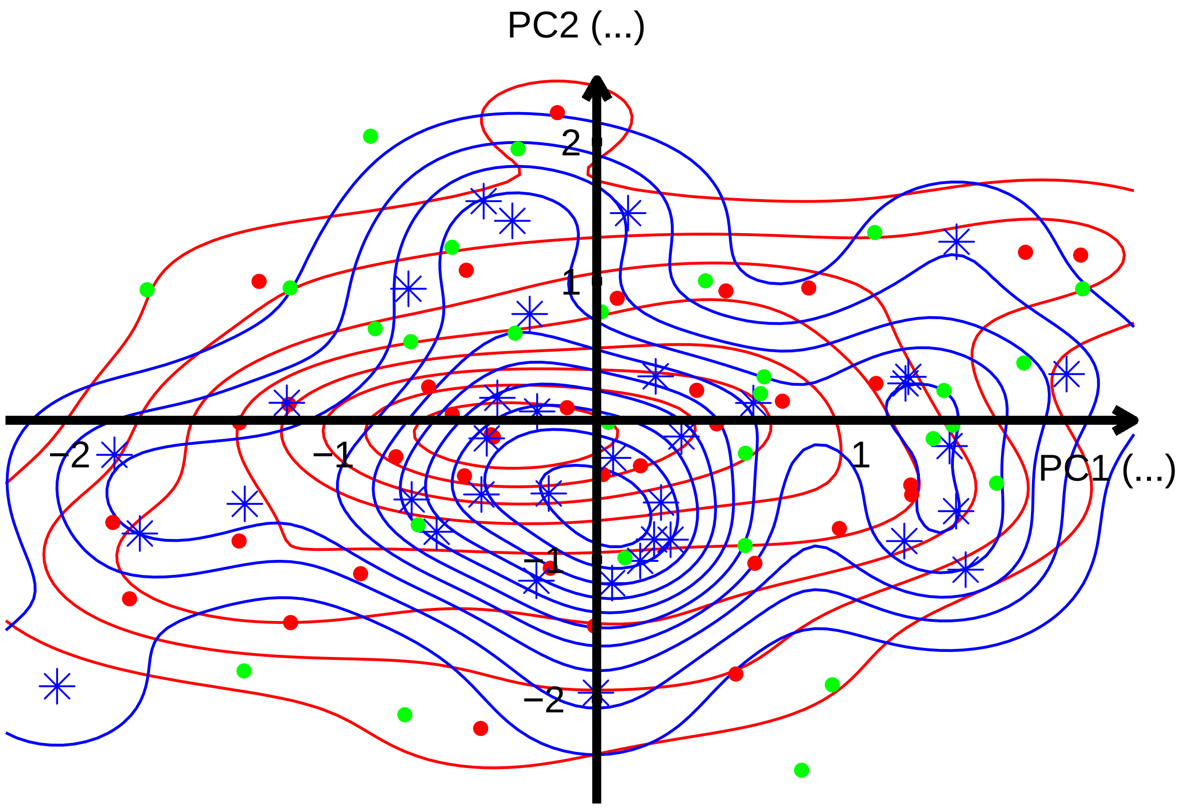
Example (Heatmap for group C):

```
shape_plot(data = df, x_col = "PC1", y_col = "PC2", group_col = "group",
           x_label = "PC1 (...)", x_label_adjust_x = -0.1, x_label_adjust_y = -0.1,
           y_label = "PC2 (...)", y_label_adjust_y =  0.4, rotate_y_label = FALSE,
           show_label_text_fields = FALSE,
           axis_linewidth = 1.5,
           group_vals = c("A","B","C"),
           point_fill = c("red","blue","green"),
           point_shape = c(21,8,21),
           point_size = c(2,4,2),
           point_color = c("red","blue","green"),
           show_heatmaps = TRUE,
           heatmap_colors = list(c("white", "red"), c("white", "blue"), c("white","green")),
           show_heatmap_for_groups = "C")
```



Example (Contours for groups A and B):

```
shape_plot(data = df, x_col = "PC1", y_col = "PC2", group_col = "group",
           x_label = "PC1 (...)", x_label_adjust_x = -0.1, x_label_adjust_y = -0.1,
           y_label = "PC2 (...)", y_label_adjust_y =  0.4, rotate_y_label = FALSE,
           show_label_text_fields = FALSE,
           axis_linewidth = 1.5,
           group_vals = c("A","B","C"),
           point_fill = c("red","blue","green"),
           point_shape = c(21,8,21),
           point_size = c(2,4,2),
           point_color = c("red","blue","green"),
           show_contours = TRUE,
           contour_colors = c("red","blue","green"),
           show_contours_for_groups = c("A","B"))
```



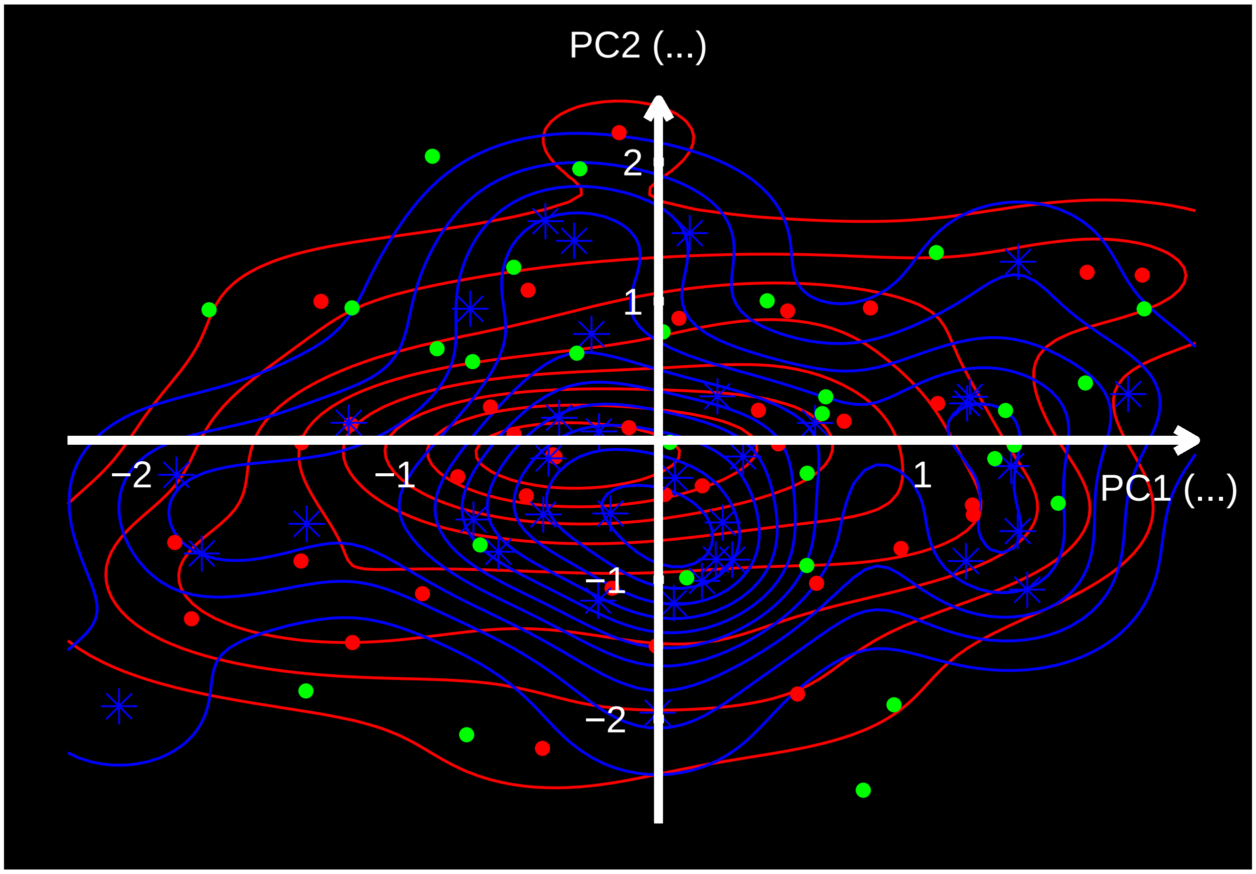
Please note, that the contour linewidth can also be adjusted with `contour_linewidth` (defaults to 0.5).

Styles

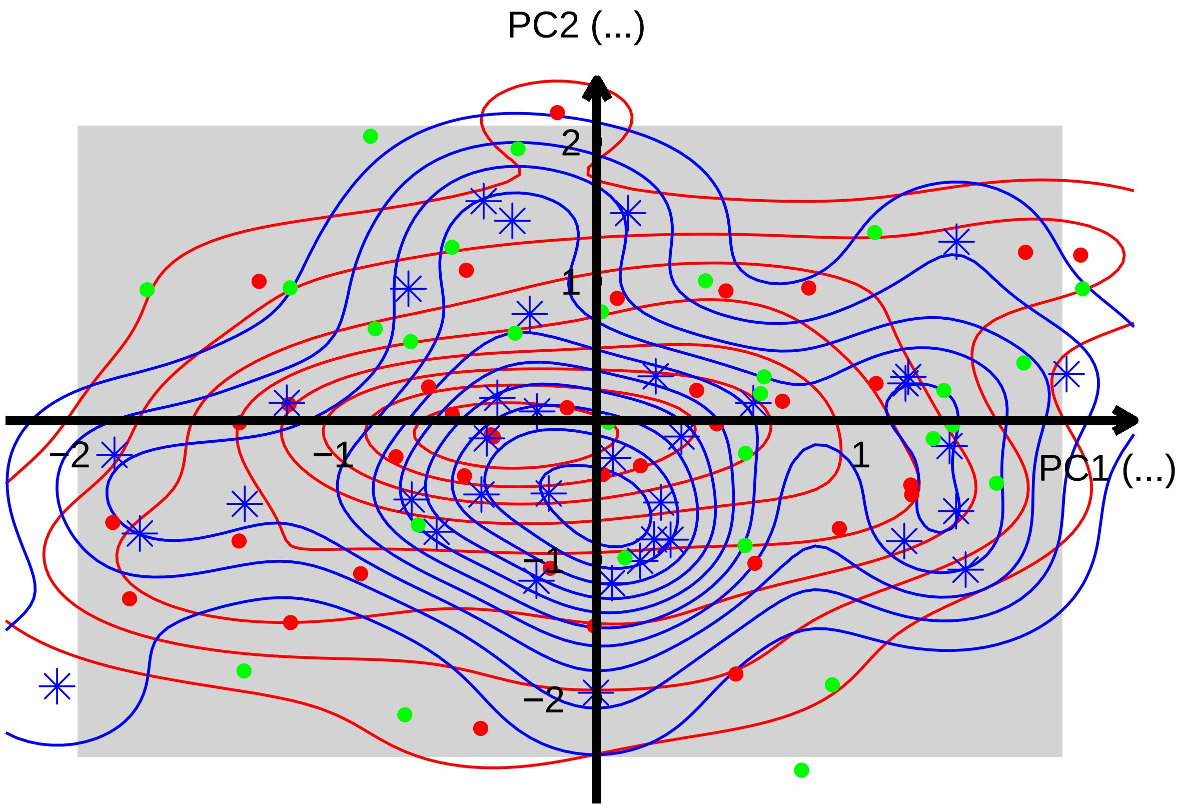
So far so good. Now we can also generate all of these plots in different styles using `plot_style =`. Right now, I have 3 styles `"Haug"`, `"inverted_Haug"`, and `"publication"`.

Inverted Haug:

```
shape_plot(data = df, x_col = "PC1", y_col = "PC2", group_col = "group",
           x_label = "PC1 (...)", x_label_adjust_x = -0.1, x_label_adjust_y = -0.1,
           y_label = "PC2 (...)", y_label_adjust_y =  0.4, rotate_y_label = FALSE,
           show_label_text_fields = FALSE,
           axis_linewidth = 1.5,
           group_vals = c("A","B","C"),
           point_fill = c("red","blue","green"),
           point_shape = c(21,8,21),
           point_size = c(2,4,2),
           point_color = c("red","blue","green"),
           show_contours = TRUE,
           contour_colors = c("red","blue","green"),
           show_contours_for_groups = c("A","B"),
           plot_style = "inverted_Haug")
```



Publication:

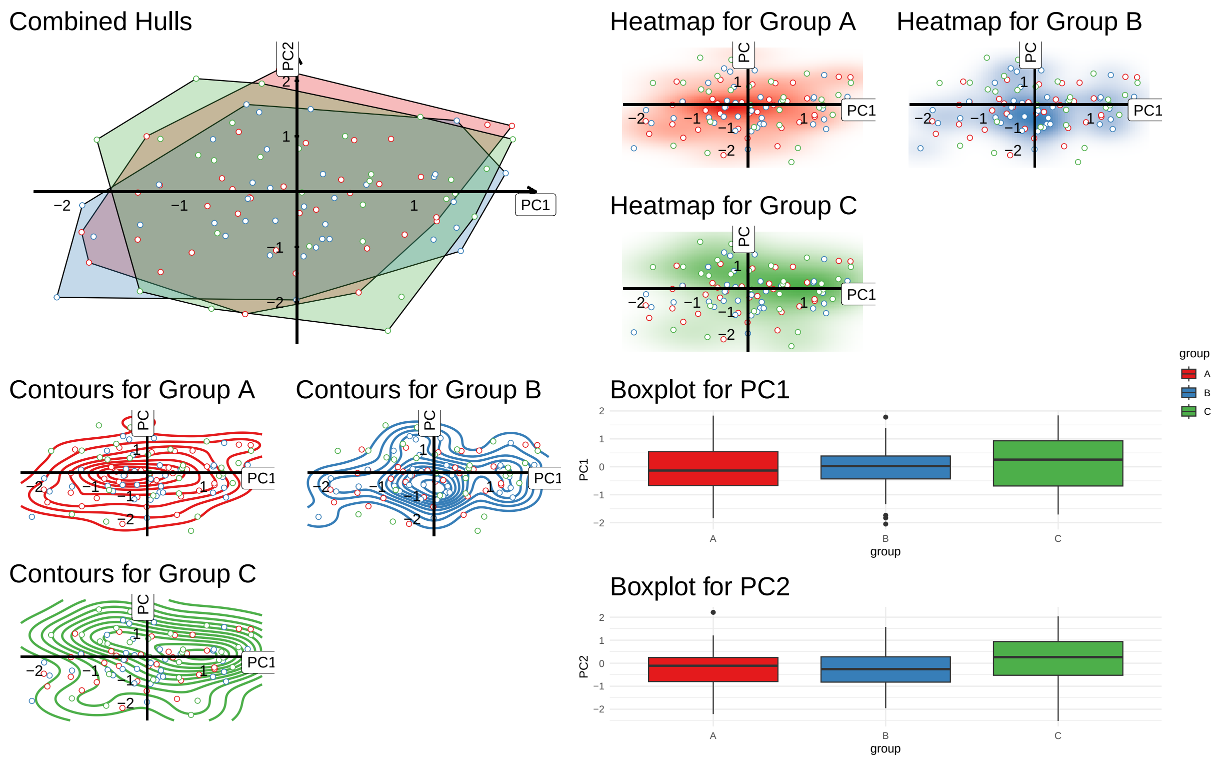


Overviews

You can easily create an overview of a new shape analysis with the function `Haug_overview()`.

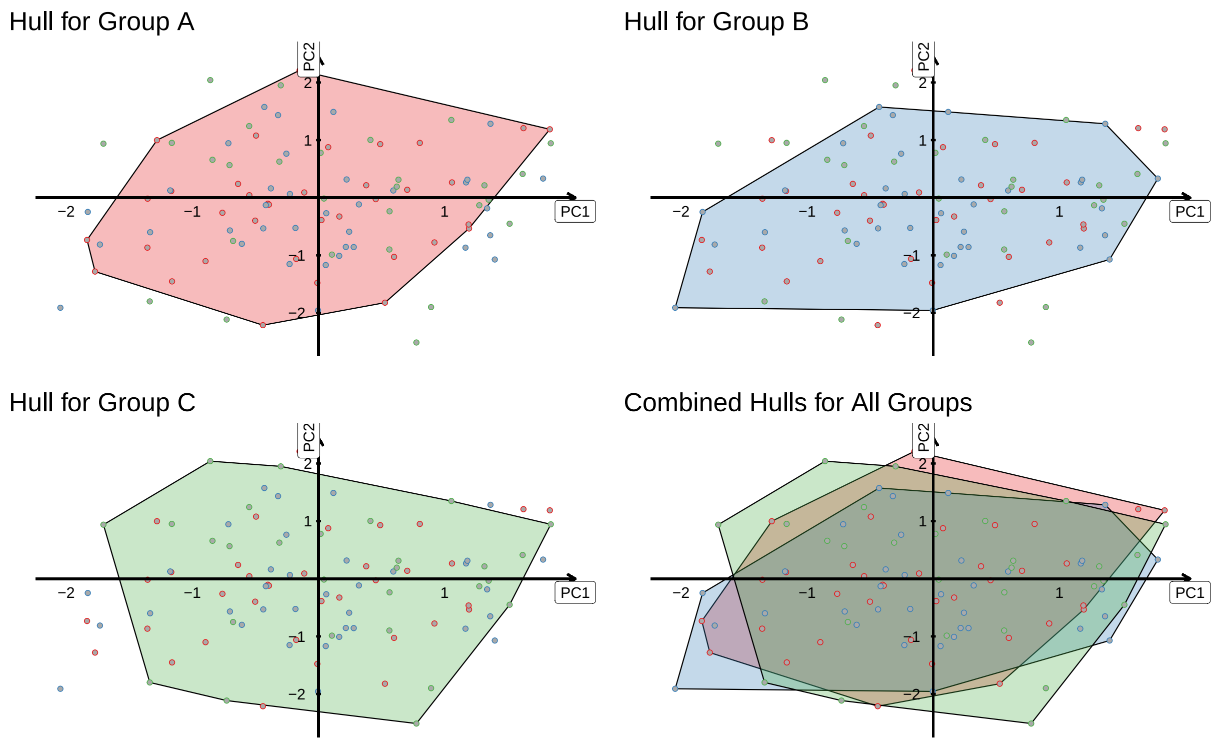
Example:

```
Haug_overview(data = df, x_col = "PC1", y_col = "PC2", group_col = "group", group_vals = c("A","B","C")
```



Or we can create panels of the different morphospaces with:

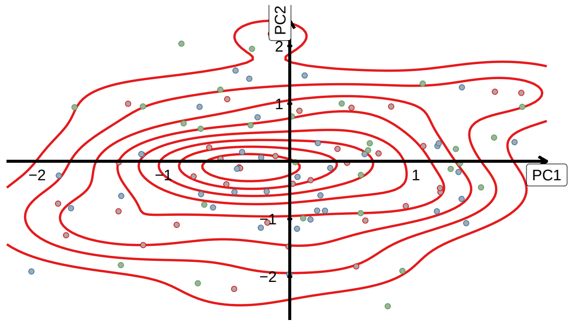
```
Haug_panel(data = df, x_col = "PC1", y_col = "PC2", group_col = "group", group_vals = c("A","B","C"))
```



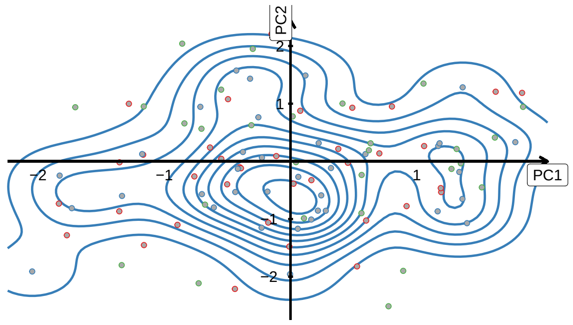
This also works with heatmaps and contours setting `panel_type = "heatmaps", "contours", or "hulls"`:

```
Haug_panel(data = df, x_col = "PC1", y_col = "PC2", group_col = "group", group_vals = c("A","B","C"), pa
```

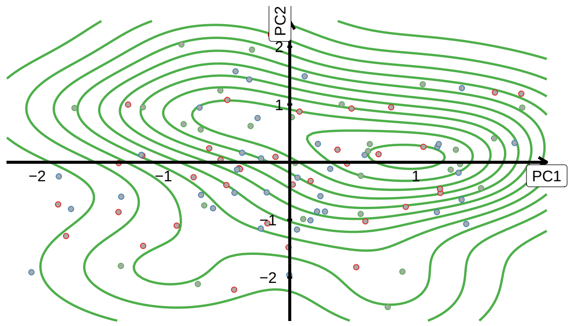
Contours for Group A



Contours for Group B



Contours for Group C


PC2
PC1