

6. PCA FACS -lab

Fay

2022-11-03

Always change the knitting directory to the working directory! # Load libraries

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.2.1
```

```
## Warning: package 'tibble' was built under R version 4.2.1
```

```
## Warning: package 'tidyr' was built under R version 4.2.1
```

```
## Warning: package 'readr' was built under R version 4.2.1
```

```
## Warning: package 'purrr' was built under R version 4.2.1
```

```
## Warning: package 'dplyr' was built under R version 4.2.1
```

```
## Warning: package 'stringr' was built under R version 4.2.1
```

```
## Warning: package 'forcats' was built under R version 4.2.1
```

```
library(dplyr)
library(stringr)
library(FactoMineR)
```

```
## Warning: package 'FactoMineR' was built under R version 4.2.1
```

```
library(reshape2)
library(corrplot)
library(factoextra)
library(lmtest)
```

```
## Warning: package 'zoo' was built under R version 4.2.1
```

```
library(ggpubr)
library(janitor)
library(pheatmap)
library(visdat)
```

Load data

```
hm <- read.csv("output_data/imputed_mice.csv")
```

vectors for selecting

```
Gene_lab <- c("IFNy", "CXCR3", "IL.6", "IL.13", "IL.10",  
             "IL1RN", "CASP1", "CXCL9", "IDO1", "IRGM1", "MPO",  
             "MUC2", "MUC5AC", "MYD88", "NCR1", "PRF1", "RETNLB", "SOCS1",  
             "TICAM1", "TNF") # "IL.12", "IRG6")  
  
#add a suffix to represent changes in data file  
Gene_lab_imp <- paste(Gene_lab, "imp", sep = "_")  
  
Genes_wild <- c("IFNy", "CXCR3", "IL.6", "IL.13", "IL.10",  
              "IL1RN", "CASP1", "CXCL9", "IDO1", "IRGM1", "MPO",  
              "MUC2", "MUC5AC", "MYD88", "NCR1", "PRF1", "RETNLB", "SOCS1",  
              "TICAM1", "TNF", "IL.12", "IRG6")  
  
Genes_wild_imp <- paste(Genes_wild, "imp", sep = "_")  
  
Facs_lab <- c("CD4", "Treg", "Div_Treg", "Treg17", "Th1",  
            "Div_Th1", "Th17", "Div_Th17", "CD8", "Act_CD8",  
            "Div_Act_CD8", "IFNy_CD4", "IFNy_CD8") #"Treg_prop", removed due to many missing va  
            #"IL17A_CD4"  
  
Facs_wild <- c("Treg", "CD4", "Treg17", "Th1", "Th17", "CD8",  
            "Act_CD8", "IFNy_CD4", "IL17A_CD4", "IFNy_CD8")
```

FACS

Lab

PCA on the lab genes -*imputed*

```
#select the genes and lab mice  
lab <- hm %>%  
  dplyr::filter(origin == "Lab", Position == "mLN") #selecting for mln to avoid  
  
# duplicates  
  
lab <- unique(lab)  
  
facs_mouse <- lab %>%  
  dplyr::select(c(Mouse_ID, all_of(Facs_lab)))
```

```
facs <- facs_mouse[, -1]

#remove rows with only nas
facs <- facs[,colSums(is.na(facs))<nrow(facs)]

#remove columns with only nas
facs <- facs[rowSums(is.na(facs)) != ncol(facs), ]

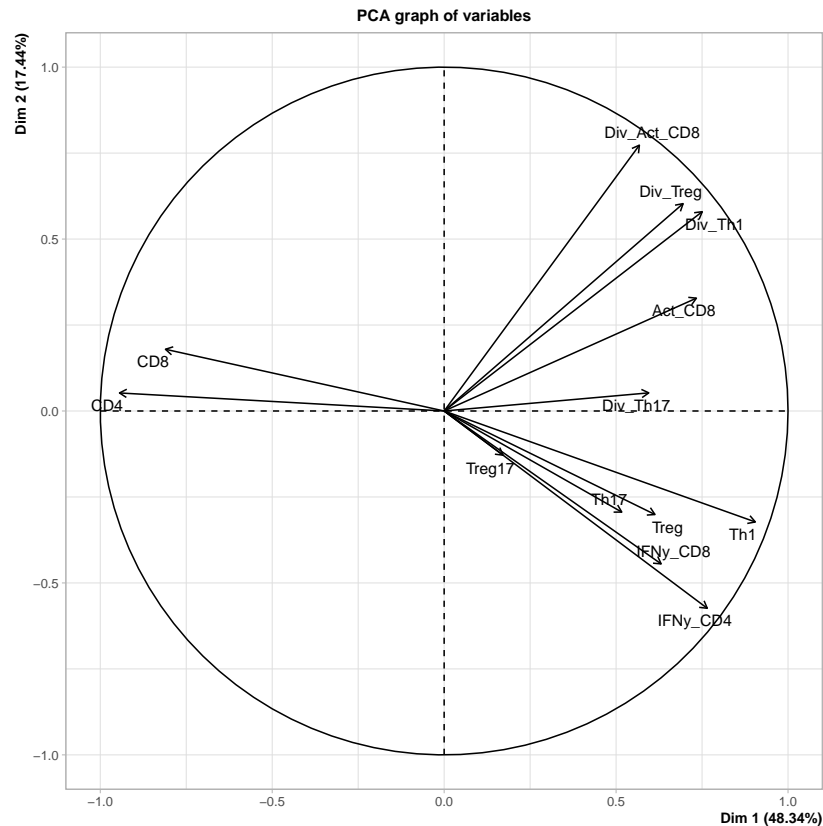
vis_dat(facs)
```

```
## Warning: 'gather_()' was deprecated in tidyr 1.2.0.
## i Please use 'gather()' instead.
## i The deprecated feature was likely used in the visdat package.
## Please report the issue at <https://github.com/ropensci/visdat/issues>.
```



```
#select same rows in the first table
facs_mouse <- facs_mouse[row.names(facs), ]

# we can now run a normal pca on the complete data set
res.pca <- PCA(facs)
```



Dimensions of the pca

Caution: When imputing data, the percentages of inertia associated with the first dimensions will be over-estimated.

Another problem: the imputed data are, when the pca is performed considered like real observations. But they are estimations!!

Visualizing uncertainty due to missing data:

→ multiple imputation: generate several plausible values for each missing data point

We here visualize the variability, that is uncertainty on the plane defined by two pca axes.

Biplot of the imputed facs pca

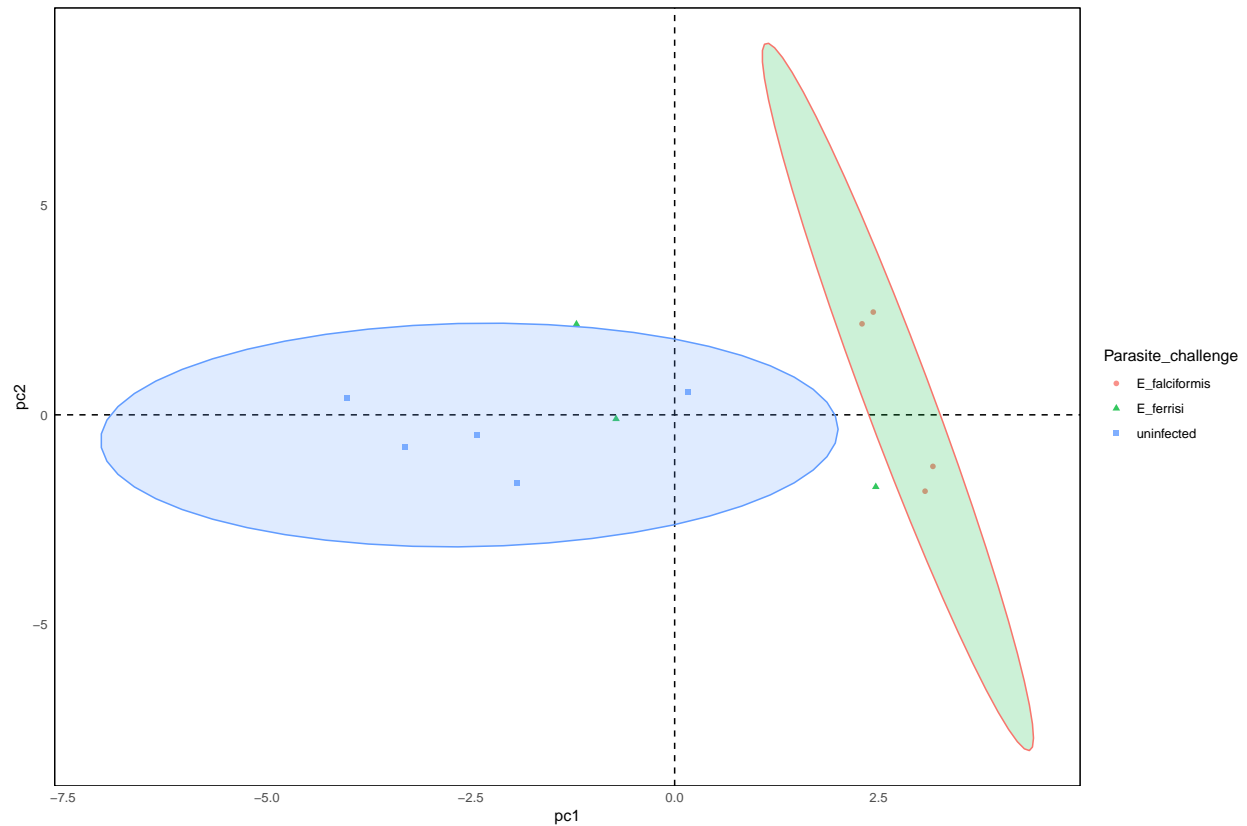
```
#Now we can make our initial plot of the PCA.
lab %>%
  ggplot(aes(x = pc1, y = pc2,
             color = Parasite_challenge,
             shape = Parasite_challenge)) +
  geom_hline(yintercept = 0, lty = 2) +
  geom_vline(xintercept = 0, lty = 2) +
  geom_point(alpha = 0.8) +
  stat_ellipse(geom="polygon",
              aes(fill = challenge_infection),
              alpha = 0.2, show.legend = FALSE,
              level = 0.95) +
  theme_minimal() +
```

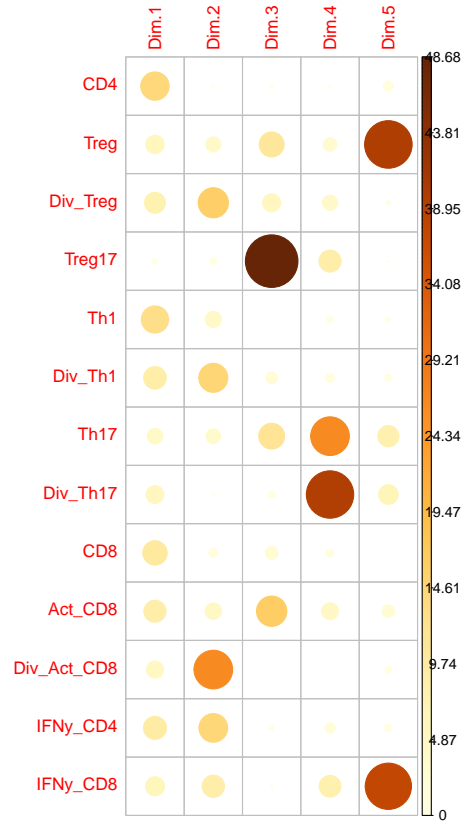
```
theme(panel.grid = element_blank(),  
      panel.border = element_rect(fill= "transparent"))
```

```
## Warning: Removed 124 rows containing non-finite values (stat_ellipse).
```

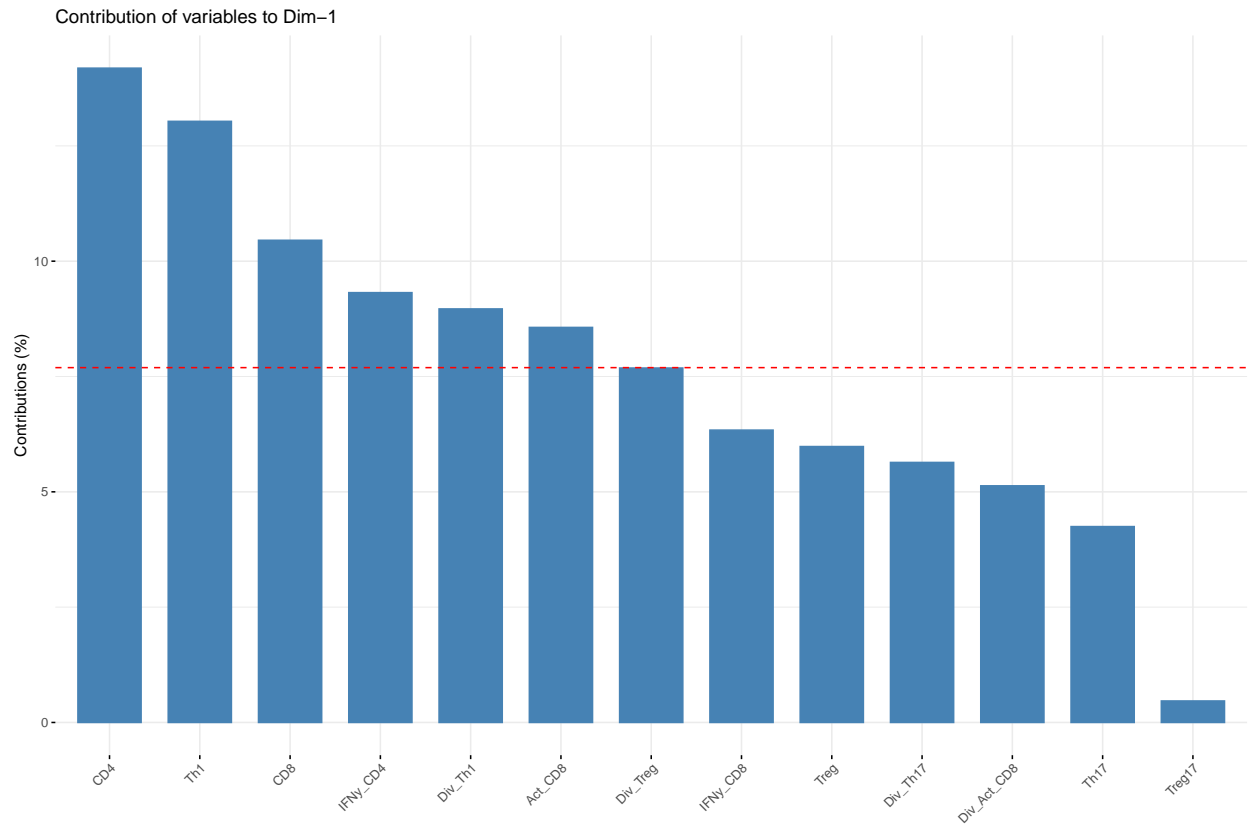
```
## Too few points to calculate an ellipse
```

```
## Warning: Removed 124 rows containing missing values (geom_point).
```

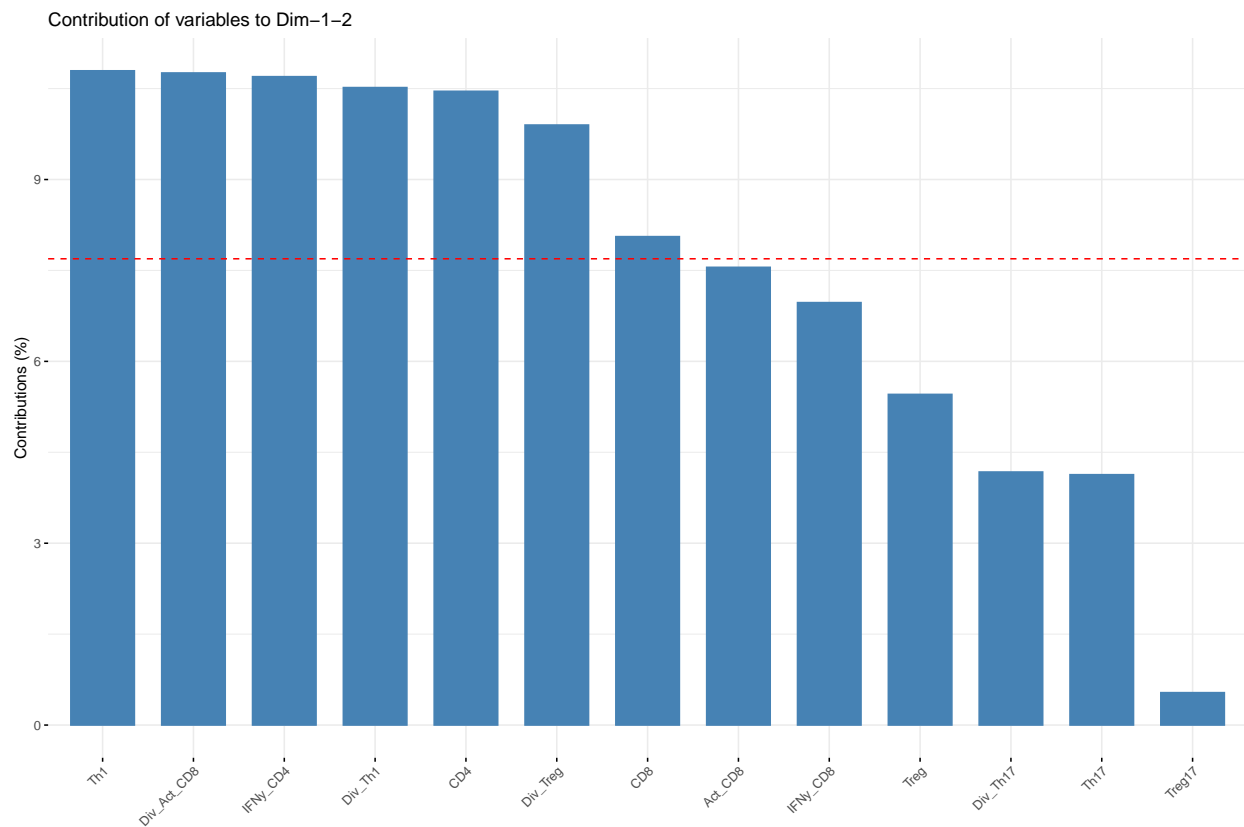
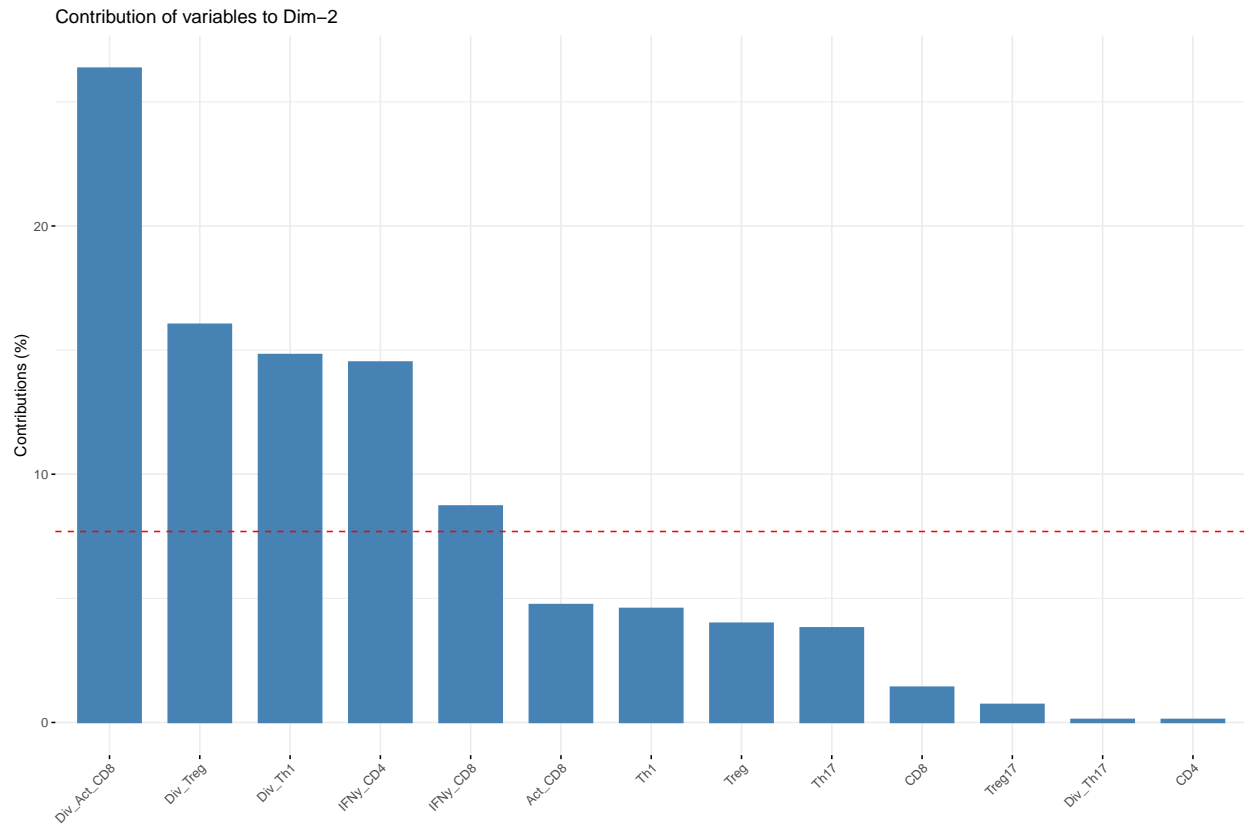




The function `fviz_contrib()` [factoextra package] can be used to draw a bar plot of variable contributions. If your data contains many variables, you can decide to show only the top contributing variables. The R code below shows the top 10 variables contributing to the principal components:



```
# Contributions of variables to PC2  
fviz_contrib(res.pca, choice = "var", axes = 2, top = 18)
```

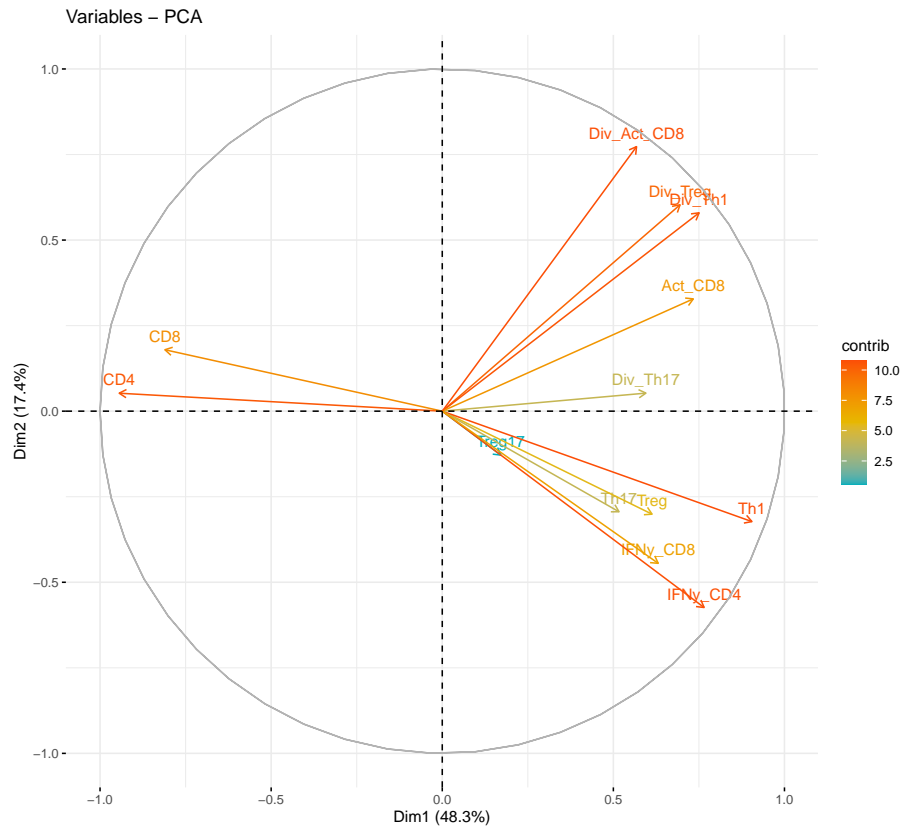


The red dashed line on the graph above indicates the expected average contribution. If the contribution of

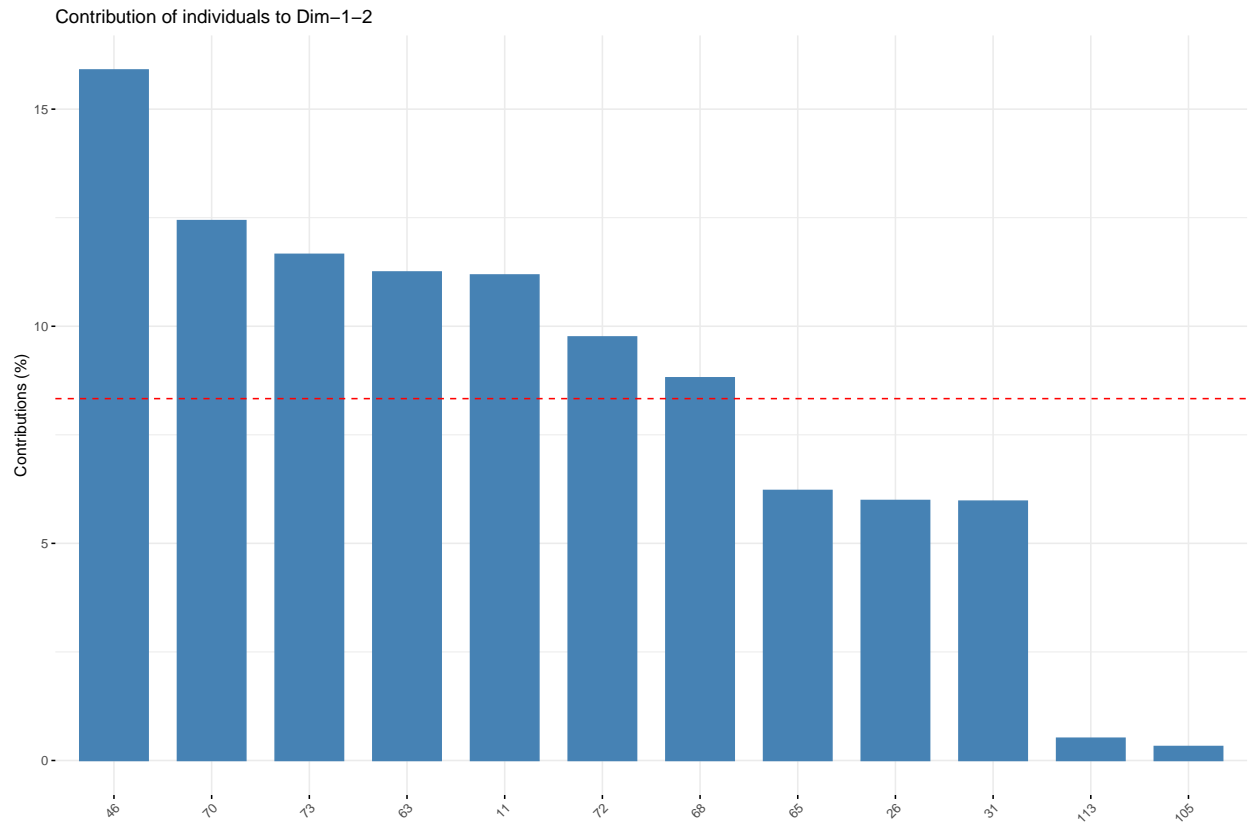
the variables were uniform, the expected value would be $1/\text{length}(\text{variables}) = 1/10 = 10\%$. For a given component, a variable with a contribution larger than this cutoff could be considered as important in contributing to the component.

Note that, the total contribution of a given variable, on explaining the variations retained by two principal components, say PC1 and PC2, is calculated as $\text{contrib} = [(C1 * \text{Eig1}) + (C2 * \text{Eig2})]/(\text{Eig1} + \text{Eig2})$, where

C1 and C2 are the contributions of the variable on PC1 and PC2, respectively Eig1 and Eig2 are the eigenvalues of PC1 and PC2, respectively. Recall that eigenvalues measure the amount of variation retained by each PC. In this case, the expected average contribution (cutoff) is calculated as follow: As mentioned above, if the contributions of the 10 variables were uniform, the expected average contribution on a given PC would be $1/10 = 10\%$. The expected average contribution of a variable for PC1 and PC2 is : $[(10 * \text{Eig1}) + (10 * \text{Eig2})]/(\text{Eig1} + \text{Eig2})$

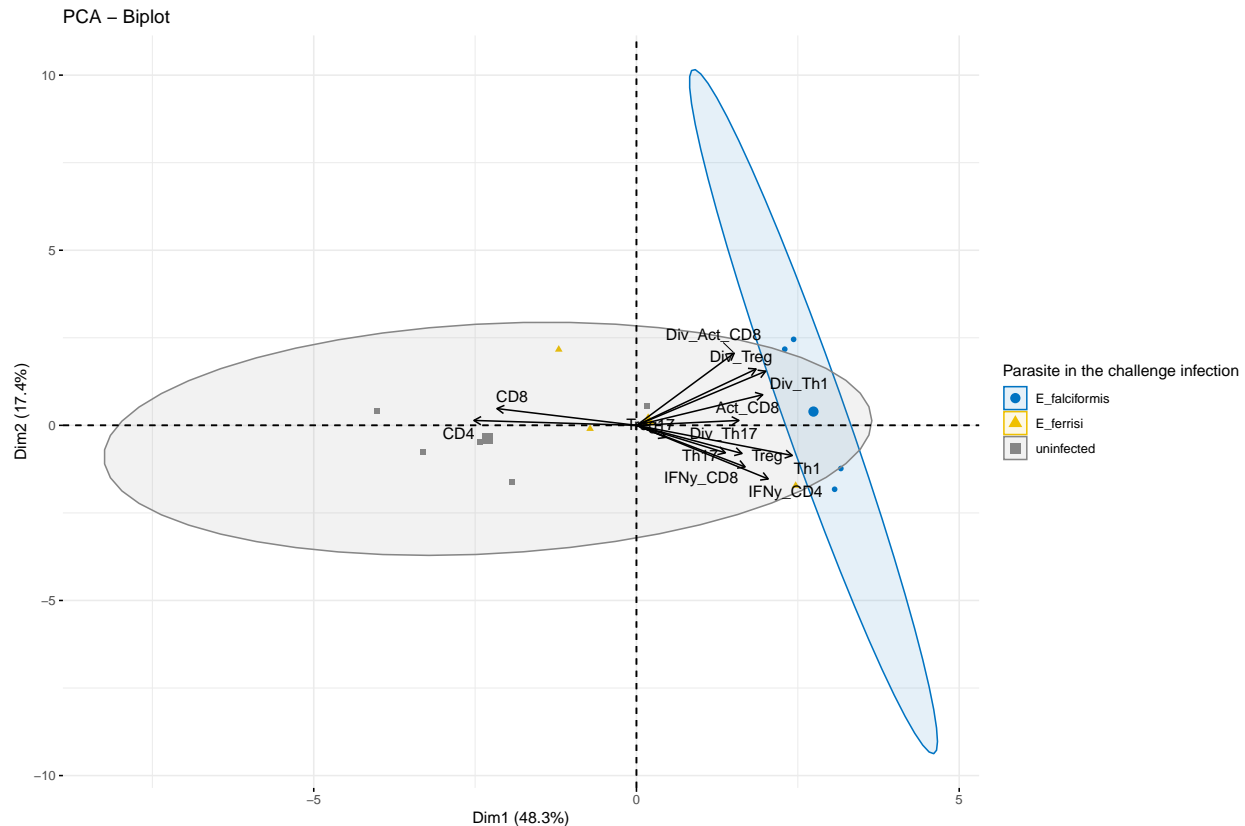


To visualize the contribution of individuals to the first two principal components:



PCA + Biplot combination

Too few points to calculate an ellipse



In the following example, we want to color both individuals and variables by groups. The trick is to use `pointshape = 21` for individual points. This particular point shape can be filled by a color using the argument `fill.ind`. The border line color of individual points is set to “black” using `col.ind`. To color variable by groups, the argument `col.var` will be used.

Linear models:

```
##
## Call:
## lm(formula = max_WL ~ pc1 + pc2 + Parasite_challenge, data = lab)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.8242 -2.2025 -0.6023  1.6168  7.1512
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    84.6516     3.5765  23.669 6.11e-08 ***
## pc1           -0.2474     1.0198  -0.243   0.815
## pc2           -1.0712     0.8686  -1.233   0.257
## Parasite_challengeE_ferrisi  3.7956     4.1134   0.923   0.387
## Parasite_challengeuninfected 10.4827     6.0884   1.722   0.129
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.043 on 7 degrees of freedom
## Multiple R-squared:  0.7724, Adjusted R-squared:  0.6423
## F-statistic: 5.938 on 4 and 7 DF, p-value: 0.02083
```

```
## [1] 73.11576

##
## Call:
## lm(formula = max_WL ~ pc1 + pc2 + Parasite_challenge + hybrid_status,
##     data = lab)
##
## Residuals:
##      11      26      31      46      63      65      68
## -9.453e-01  3.104e+00 -3.104e+00  1.388e-16  7.155e+00  2.389e+00 -2.389e+00
##      70      72      73     105     113
## -3.800e+00  1.503e-01 -3.505e+00  1.661e+00 -7.156e-01
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      83.6053      6.7657  12.357 0.000246 ***
## pc1              -0.1753      1.6401  -0.107 0.920008
## pc2             -1.0498      1.2483  -0.841 0.447680
## Parasite_challengeE_ferrisi  4.2503      6.0665   0.701 0.522150
## Parasite_challengeuninfected 11.1185      8.4780   1.311 0.259915
## hybrid_statusF1 hybrid    0.8879      5.1647   0.172 0.871855
## hybrid_statusF1 M. m. musculus 1.1953      7.1414   0.167 0.875199
## hybrid_statusother    0.8402      5.6015   0.150 0.888028
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.31 on 4 degrees of freedom
## Multiple R-squared:  0.7756, Adjusted R-squared:  0.383
## F-statistic: 1.975 on 7 and 4 DF,  p-value: 0.2663

## [1] 78.94354
```

Try instead: LLR test (likelihood ration) (LM4 package)?

<https://www.rdocumentation.org/packages/lmtest/versions/0.9-38/topics/lrtest>

In this way you compare each model, with the different variables used to predict.

Another way is to compare the AIC. (function : step)

```
weight_lm3 <- lm(max_WL ~ pc1 + pc2 + hybrid_status, data = lab)
weight_no_pc1 <- lm(max_WL ~ pc2 + hybrid_status, data = lab)
weight_no_pc2 <- lm(max_WL ~ pc1 + hybrid_status, data = lab)
weight_no_hybrid <- lm(max_WL ~ pc1 + pc2, data = lab)
lrtest(weight_lm3, weight_no_pc1)
```

```
## Likelihood ratio test
##
## Model 1: max_WL ~ pc1 + pc2 + hybrid_status
## Model 2: max_WL ~ pc2 + hybrid_status
##   #Df LogLik Df  Chisq Pr(>Chisq)
## 1    7 -32.853
## 2    6 -35.217 -1  4.7271   0.02969 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
lrtest(weight_lm3, weight_no_pc2)
```

```
## Likelihood ratio test
##
## Model 1: max_WL ~ pc1 + pc2 + hybrid_status
## Model 2: max_WL ~ pc1 + hybrid_status
##   #Df LogLik Df  Chisq Pr(>Chisq)
## 1    7 -32.853
## 2    6 -34.899 -1 4.0916    0.0431 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
lrtest(weight_lm3, weight_no_hybrid)
```

```
## Likelihood ratio test
##
## Model 1: max_WL ~ pc1 + pc2 + hybrid_status
## Model 2: max_WL ~ pc1 + pc2
##   #Df LogLik Df  Chisq Pr(>Chisq)
## 1    7 -32.853
## 2    4 -32.871 -3 0.0354    0.9982

##
## Call:
## lm(formula = max_WL ~ pc1 + pc2 + hybrid_status, data = lab)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.4546 -2.7120 -0.4155  2.0890  6.8441
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      90.1958     3.3487  26.934 1.73e-07 ***
## pc1              -1.8640     1.0952  -1.702   0.140
## pc2              -1.7298     1.1079  -1.561   0.169
## hybrid_statusF1 hybrid    -0.5130     5.0099  -0.102   0.922
## hybrid_statusF1 M. m. musculus -0.7903     6.8983  -0.115   0.913
## hybrid_statusother    -0.1524     5.1742  -0.029   0.977
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.288 on 6 degrees of freedom
## Multiple R-squared:  0.6663, Adjusted R-squared:  0.3882
## F-statistic: 2.396 on 5 and 6 DF,  p-value: 0.1587

## [1] 79.70627

##
## Call:
## lm(formula = max_WL ~ pc1 + pc2 + infection_history, data = lab)
##
```

```

## Residuals:
##      11      26      31      46      63      65      68
##  8.327e-17  4.341e-18 -3.242e+00  4.956e-01  1.150e-16  1.814e+00 -7.167e-01
##      70      72      73     105     113
## -9.191e-16  1.782e+00 -1.782e+00  9.323e-01  7.167e-01
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      79.9799      3.9748  20.122 0.000268
## pc1             -0.1568      0.7571  -0.207 0.849226
## pc2             -1.3830      1.3564  -1.020 0.382963
## infection_historyfalciformis_ferrisi      6.4245      3.7844      1.698 0.188145
## infection_historyfalciformis_uninfected  13.6060      6.0317      2.256 0.109355
## infection_historyferrisi_falciformis     11.1515      3.9433      2.828 0.066300
## infection_historyferrisi_ferrisi        12.6122      6.7093      1.880 0.156724
## infection_historyferrisi_uninfected     15.6525      4.9222      3.180 0.050095
## infection_historyuninfected_falciformis   3.5155      6.4287      0.547 0.622535
##
## (Intercept)          ***
## pc1
## pc2
## infection_historyfalciformis_ferrisi
## infection_historyfalciformis_uninfected
## infection_historyferrisi_falciformis      .
## infection_historyferrisi_ferrisi
## infection_historyferrisi_uninfected      .
## infection_historyuninfected_falciformis
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.726 on 3 degrees of freedom
## Multiple R-squared:  0.9556, Adjusted R-squared:  0.8374
## F-statistic:  8.08 on 8 and 3 DF,  p-value: 0.05655

## [1] 61.48955

##
## Call:
## lm(formula = max_WL ~ pc1 + pc2, data = lab)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5.574 -2.440 -0.603  1.995  7.083
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   89.9682      1.2482  72.080 9.63e-14 ***
## pc1           -1.8205      0.4979  -3.656 0.00527 **
## pc2           -1.7633      0.8289  -2.127 0.06231 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.324 on 9 degrees of freedom
## Multiple R-squared:  0.6653, Adjusted R-squared:  0.591

```

```
## F-statistic: 8.946 on 2 and 9 DF, p-value: 0.007257
```

```
##           df      AIC
## weight_lm      6 73.11576
## weight_lm_exp_only 4 73.74164
```

repeating the heatmap on the now imputed data

```
# turn the data frame into a matrix and transpose it. We want to have each cell
# type as a row name
facs_mouse <- t(as.matrix(facs_mouse))

# turn the first row into column names
facs_mouse %>%
  row_to_names(row_number = 1) -> heatmap_data

heatmap_data <- as.data.frame(heatmap_data)

table(rowSums(is.na(heatmap_data)) == nrow(heatmap_data))
```

```
##
## FALSE
##    13
```

```
# turn the columns to numeric other wise the heatmap function will not work
heatmap_data[] <- lapply(heatmap_data, function(x) as.numeric(as.character(x)))

# remove columns with only NAs
heatmap_data <- Filter(function(x) !all(is.na(x)), heatmap_data)

# remove rows with only NAs
heatmap_data <- heatmap_data[, colSums(is.na(heatmap_data)) !=
  nrow(heatmap_data)]

# Prepare the annotation data frame
annotation_df <- as_tibble(lab) %>%
  dplyr::select(c("Mouse_ID", "Parasite_challenge", "infection_history",
    "mouse_strain", "max_WL"))

annotation_df <- unique(annotation_df)

annotation_df <- as.data.frame(annotation_df)

### Prepare the annotation columns for the heatmap
rownames(annotation_df) <- annotation_df$Mouse_ID
```

```
# Match the row names to the heatmap data frame
rownames(annotation_df) <- colnames(heatmap_data)

#remove the unnecessary column
annotation_df <- annotation_df %>% dplyr::select(-Mouse_ID, )
```

Heatmap on facs expression data:

