Step 1. Loading file:

inputdata <- read.csv("C:/Users/altera/Downloads/Data Biofilm.csv",header = TRUE)

CODE	R	I	S	Biofilm
H1	3	2	9	0.204542311
H2	1	5	8	0.545542311
H3	1	0	13	-0.007457689
H4	0	0	14	0.046542311
H5	0	1	13	-0.020457689

Checking input data

str(inputdata)

```
'data.frame':     72 obs. of  5 variables:
 $ CODE   : Factor w/ 72 levels "A1","A10","A11",..: 27 38 49 50 51 52 53 54 55 28 ...
 $ R      : int  3 1 1 0 0 1 0 0 1 1 ...
 $ I      : int  2 5 0 0 1 2 0 0 0 0 ...
 $ S      : int  9 8 13 14 13 11 14 14 13 13 ...
 $ Biofilm: num  0.20454 0.54554 -0.00746 0.04654 -0.02046 ...
```

Convert factor data for column 1

inputdata$CODE <- as.factor(inputdata$CODE)
str(inputdata)

```
'data.frame':     72 obs. of  5 variables:
 $ CODE   : Factor w/ 72 levels "A1","A10","A11",..: 27 38 49 50 51 52 53 54 55 28 ...
 $ R      : int  3 1 1 0 0 1 0 0 1 1 ...
 $ I      : int  2 5 0 0 1 2 0 0 0 0 ...
 $ S      : int  9 8 13 14 13 11 14 14 13 13 ...
 $ Biofilm: num  0.20454 0.54554 -0.00746 0.04654 -0.02046 ...
```

Step 2. Compute the Principal Components

mtpca.pca <- prcomp(inputdata, center = TRUE,scale. = TRUE)
OR: specific columns: mtpca.pca <- prcomp(inputdata[,c(2:5)], center = TRUE,scale. = TRUE)

Then you can have a peek at your PCA object with summary()

summary(mtpca.pca)

```
Importance of components:
                          PC1    PC2    PC3      PC4
Standard deviation     1.5503 0.9636 0.8173 1.35e-16
Proportion of Variance 0.6009 0.2321 0.1670 0.00e+00
Cumulative Proportion  0.6009 0.8330 1.0000 1.00e+00
```

Let's call str() to have a look at your PCA object.

str(mtpca.pca)

```
List of 5
 $ sdev    : num [1:4] 1.55 9.64e-01 8.17e-01 1.35e-16
 $ rotation: num [1:4, 1:4] -0.52861 -0.50882 0.63804 -0.23362 0.00696
...
```

```
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:4] "R" "I" "S" "Biofilm"
  .. ..$ : chr [1:4] "PC1" "PC2" "PC3" "PC4"
 $ center  : Named num [1:4] 2.26 2.32 9.42 0.21
  ..- attr(*, "names")= chr [1:4] "R" "I" "S" "Biofilm"
 $ scale   : Named num [1:4] 1.891 1.775 2.982 0.263
  ..- attr(*, "names")= chr [1:4] "R" "I" "S" "Biofilm"
 $ x       : num [1:72, 1:4] -0.199 -1.017 1.978 2.423 1.982 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : chr [1:4] "PC1" "PC2" "PC3" "PC4"
 - attr(*, "class")= chr "prcomp
```

Step 4. Plotting PCA

You will use the ggbiplot package, which offers a user-friendly and pretty function to plot biplots. A biplot is a type of plot that will allow you to visualize how the samples relate to one another in our PCA (which samples are similar and which are different) and will simultaneously reveal how each variable contributes to each principal component.

Install library

library(devtools)
install_github("vqv/ggbiplot")

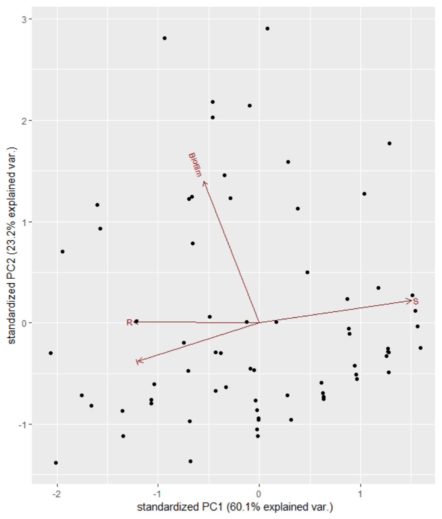
Next, you can call ggbiplot on your PCA:

#library(ggbiplot)
#ggbiplot(mtpca.pca)

Next, you can call ggbiplot on your PCA:

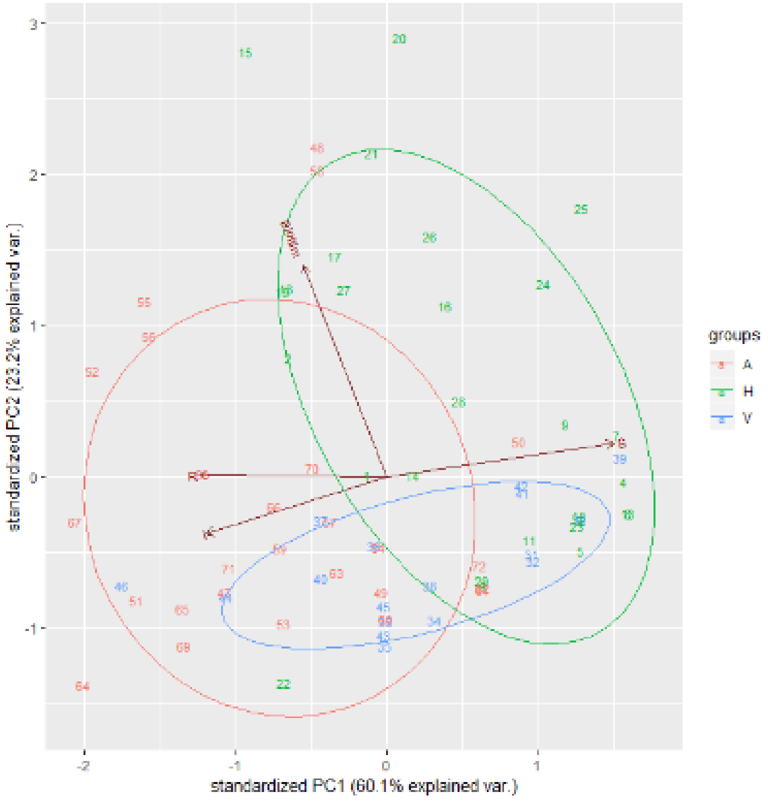
library(ggbiplot)
ggbiplot(mtpca.pca)

```
Loading required package: ggplot2
Find out what's changed in ggplot2 at
https://github.com/tidyverse/ggplot2/releases.
Loading required package: plyr
Loading required package: scales
Loading required package: grid
```



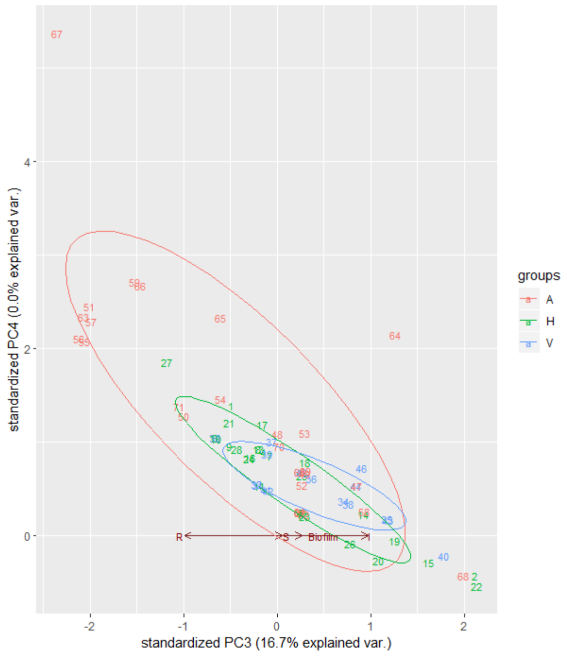
Step 5. Interpreting the results

mtpca.sample <- c(rep("H",29),rep("V",17),rep("A",26))
ggbiplot(mtpca.pca,ellipse=TRUE, labels=rownames(inputdata), groups=mtpca.sample)



Step 6. Let's have a look at PC3 and PC4:

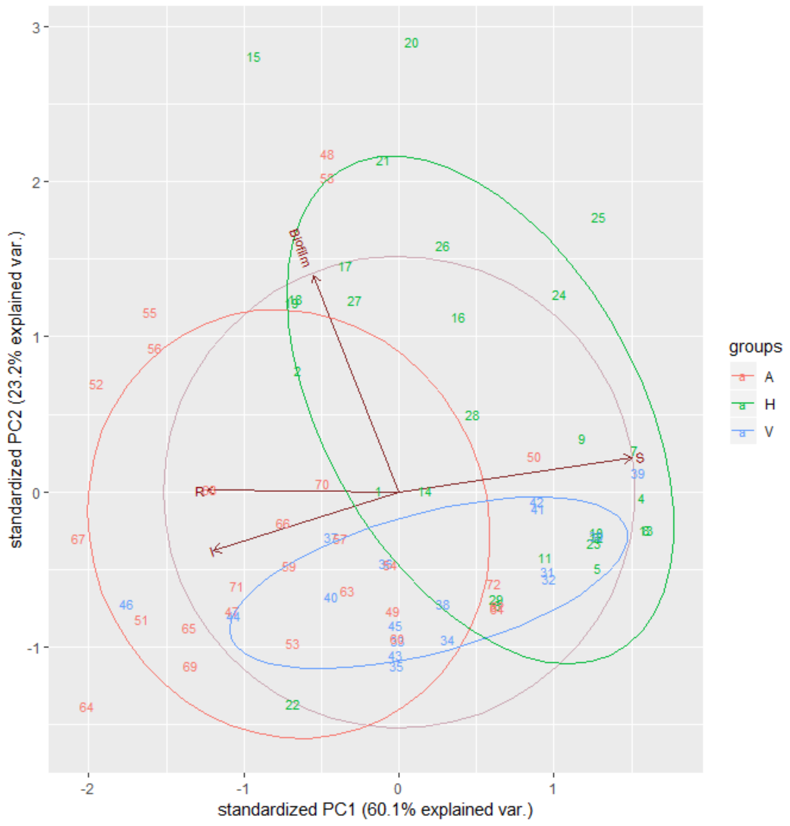
ggbiplot(mtpca.pca,ellipse=TRUE,choices=c(3,4),labels=rownames(inputdata), groups=mtpca.sample)



Step 7. Graphical parameters with ggbiplot

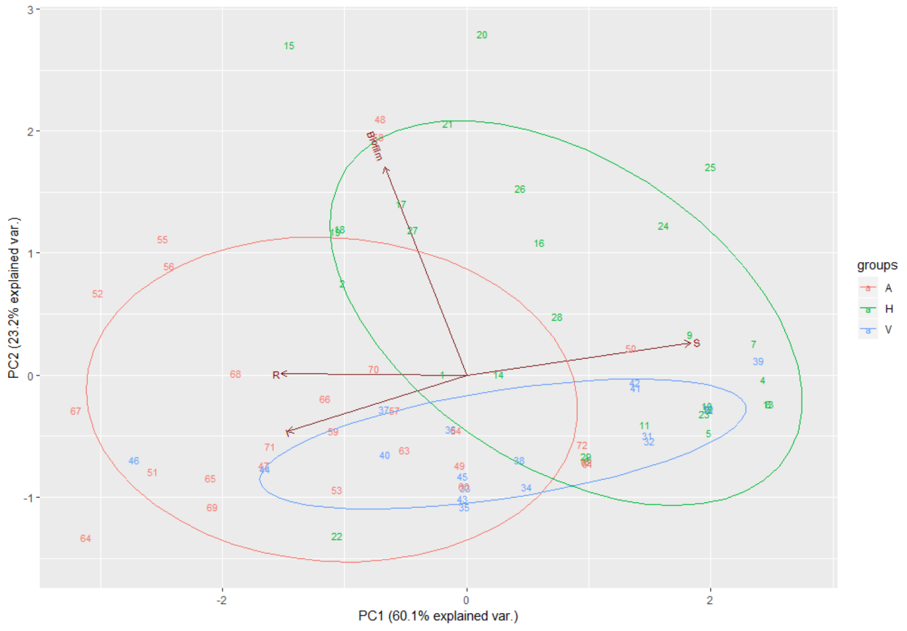
There are also some other variables you can play with to alter your biplots. You can add a circle to the center of the dataset (circle argument):

ggbiplot(mtpca.pca,ellipse=TRUE,circle=TRUE, labels=rownames(inputdata), groups=mtpca.sample)



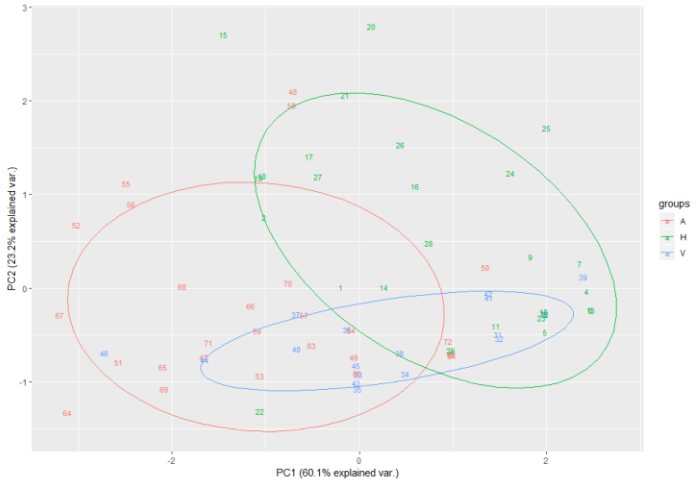
Step 8. You can also scale the samples (obs.scale) and the variables (var.scale):

ggbiplot(mtpca.pca,ellipse=TRUE,obs.scale = 1, var.scale = 1, labels=rownames(inputdata), groups=mtpca.sample)



Step 9. You can also remove the arrows altogether, using var.axes.

ggbiplot(mtpca.pca,ellipse=TRUE,obs.scale = 1, var.scale = 1,var.axes=FALSE,   labels=rownames(inputdata), groups=mtpca.sample)

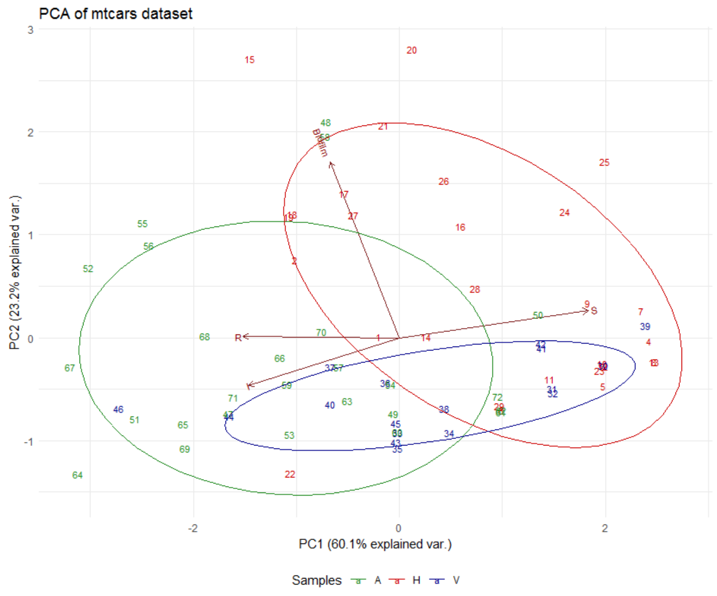


Step 10. Customize ggbiplot

As ggbiplot is based on the ggplot function, you can use the same set of graphical parameters to alter your biplots as you would for any ggplot. Here, you're going to:

Specify the colours to use for the groups with scale_colour_manual()
Add a title with ggtitle()
Specify the minimal() theme
Move the legend with theme()

ggbiplot(mtpca.pca,ellipse=TRUE,obs.scale = 1, var.scale = 1, labels=rownames(inputdata), groups=mtpca.sample) +
scale_colour_manual(name="Samples", values= c("forest green", "red3", "dark blue"))+
ggtitle("PCA of mtcars dataset")+
theme_minimal()+
theme(legend.position = "bottom")


PCA of mtcars dataset
PC1 (60.1% explained var.)
PC2 (23.2% explained var.)
Samples
A
H
V
R
S