# Class 7 Lab

Dylan Mullaney (A16869792)

## Hands on with Principal Component Analysis

Examine a 17-dimensional data detailing food consumption in England, Wales, Scotland, and Northern Ireland.

Read the provide input file

> Q1. How many rows and columns are in your new data frame named x? What R functions could you use to answer this questions?

17 rows, 5 columns, using nrow() and ncol()

```
url<- "https://tinyurl.com/UK-foods"
x <- read.csv(url) ##To remove numbers from row, could add ', row.names=1' after url
nrow(x)
```

```
[1] 17
```

```
ncol(x)
```

```
[1] 5
```

```
head(x)
```

```
              X England Wales Scotland N.Ireland
1          Cheese     105   103      103        66
2   Carcass_meat     245   227      242       267
3     Other_meat     685   803      750       586
4           Fish     147   160      122        93
5 Fats_and_oils     193   235      184       209
6         Sugars     156   175      147       139
```

```
#View(x)
```

Remove the first column so it starts with the value of each food and not a number

```
rownames(x) <- x[,1]
x <- x[,-1]
head(x)
```

```
               England Wales Scotland N.Ireland
Cheese             105   103      103        66
Carcass_meat       245   227      242       267
Other_meat         685   803      750       586
Fish               147   160      122        93
Fats_and_oils      193   235      184       209
Sugars             156   175      147       139
```

```
dim(x)
```

```
[1] 17  4
```

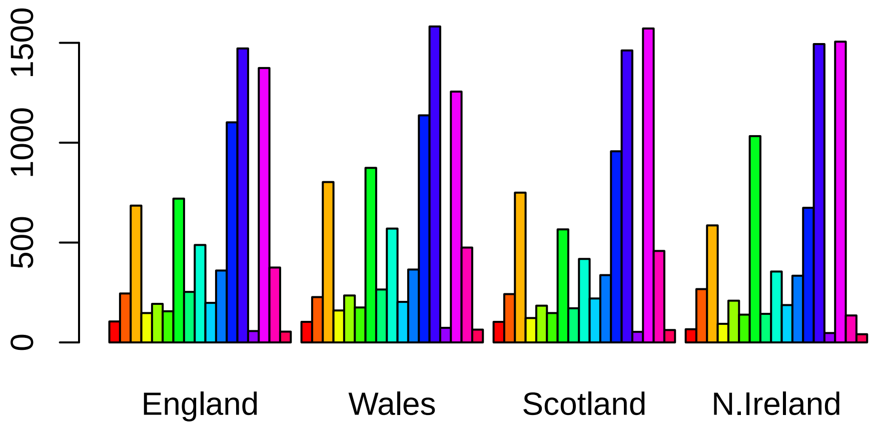
```
#DO NOT run again, will remove another column, can be fixed by rerunning code above so that
```

> Q2. Which approach to solving the 'row-names problem' mentioned above do you prefer and why? Is one approach more robust than another under certain circumstances?

The minus indexing technique can be tricky because it requires that you only run functional code once or else it will continue to remove values from the dataset. For this reason, I think the technique below might be more functional. `x <- read.csv(url, row.names=1)`

Make a fun rainbow plot of this data

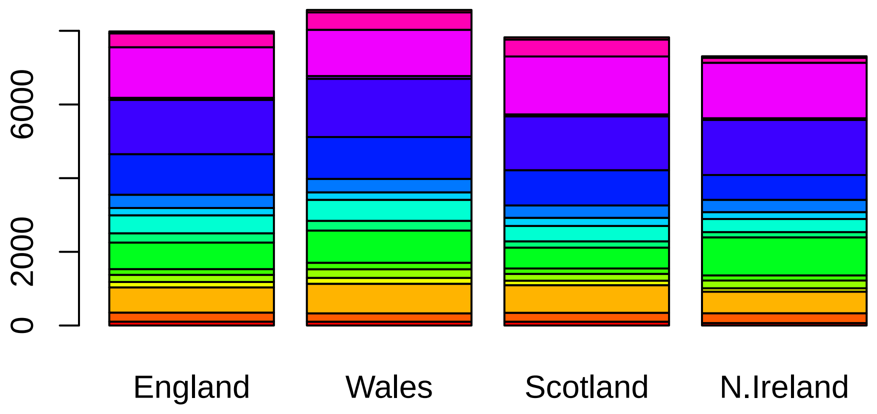
```
barplot(as.matrix(x), beside=T, col=rainbow(nrow(x)))
```



Q3: Changing what optional argument in the above barplot() function results in the following plot?

Setting 'beside' as equal to "F". The beside line of code changes the arrangement of the bars in the plot.

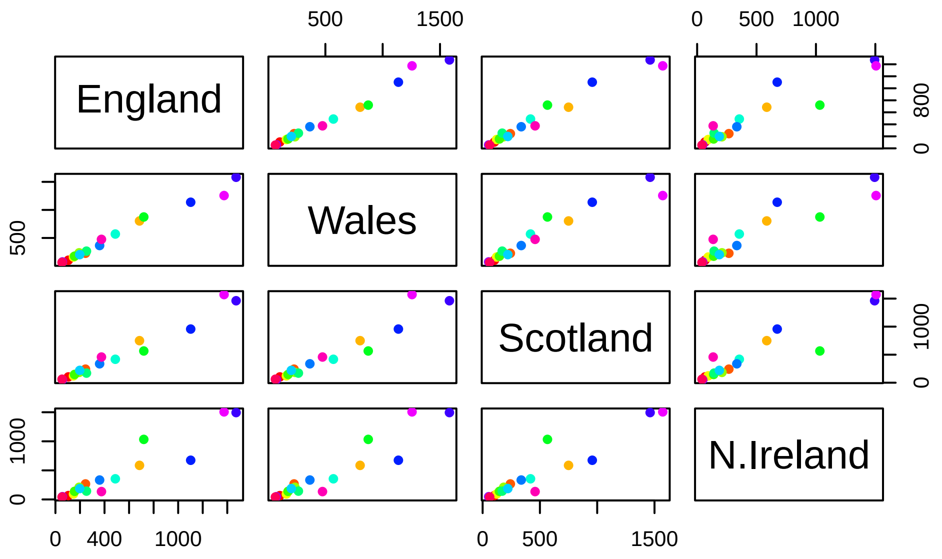
```
barplot(as.matrix(x), beside=F, col=rainbow(nrow(x)))
```



> Q5: Generating all pairwise plots may help somewhat. Can you make sense of the following code and resulting figure? What does it mean if a given point lies on the diagonal for a given plot?

This kind of display is only workable with small datasets and is called a scatterplot matrix. Each variable is listed in a line and plotted against each other. In the first row, England is the y axis. The x axis depends on the other intersecting country. So, row 1 is England as y axis and column 1 is England as x axis. If a point lies on a diagonal, it means that there is a perfect correlation between the two variables, ie they are both the same value.

```r
pairs(x, col=rainbow(nrow(x)), pch=16)
```



Looking at these types of "pairwise plots" can be helpful but it does not scale well and kind of sucks (time consuming, laborious, error prone)!

> Q6. What is the main differences between N. Ireland and the other countries of the UK in terms of this data-set?

N. Ireland has a noticeable difference in the bright green dot towards the center of the dataset, as it lies astray from the diagonal line. However, it is hard to deduce much specifically with so many variables and different plots.

## PCA to the rescue

The main function for PCA in base R is called 'prcomp()'. This function wants the transpose of our input data - i.e. the important food categories in as columns and the countries as rows.

```
pca <- prcomp (t(x))
summary (pca)
```

```
Importance of components:
                            PC1      PC2      PC3       PC4
Standard deviation     324.1502 212.7478 73.87622 2.921e-14
Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

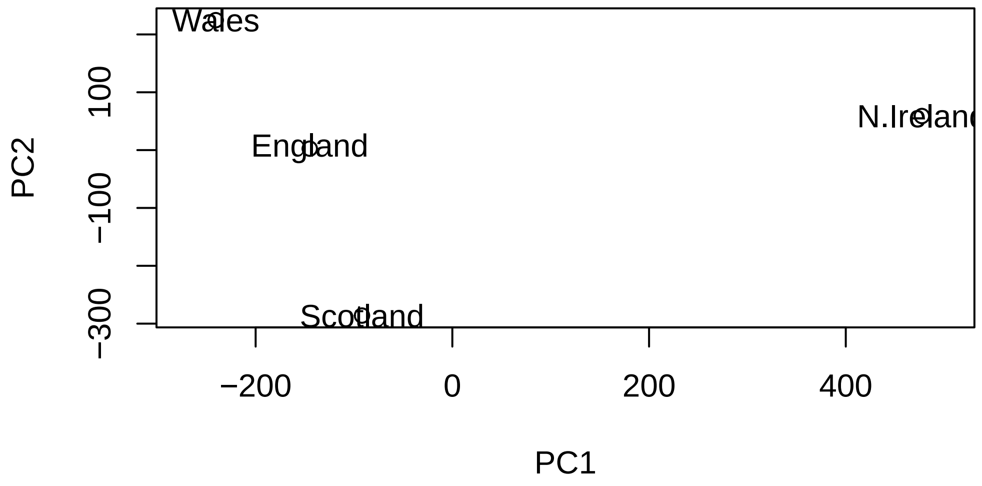
The ‘pca$x’ result object is where we will focus first as this details how the countries are related to each other in terms of our new “axis” (ie “PCs”, “eigenvectors”, etc.).

```
head(pca$x)
```

```
                PC1         PC2        PC3           PC4
England   -144.99315   -2.532999 105.768945 -9.152022e-15
Wales     -240.52915 -224.646925 -56.475555  5.560040e-13
Scotland   -91.86934  286.081786 -44.415495 -6.638419e-13
N.Ireland  477.39164  -58.901862  -4.877895  1.329771e-13
```

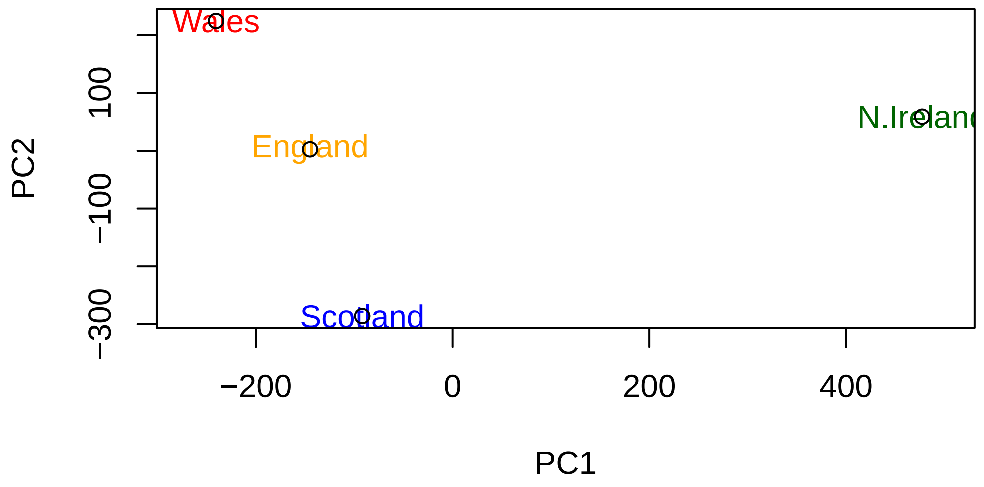
Q7. Complete the code below to generate a plot of PC1 vs PC2. The second line adds text labels over the data points. plot(pca$x[, 1], pcax$ [,2], pch=16, col=c(“orange”, “red”, “blue”, “darkgreen”), xlab=“PC1”, ylab= “PC2”, xlim=c(-270,500), text(pca$x[, 1], pcax$[,2], colnames(x)))

```
pca$x [,2] <- -pca$x [,2]
#PCA1 is correct, PCA2 is opposite for some reason? Y axis needs to be mult by -1
plot(pca$x[, 1], pca$x[, 2], xlab="PC1", ylab="PC2", xlim=c(-270,500))
text(pca$x[, 1], pca$x[, 2], colnames(x))
```



Q8. Customize your plot so that the colors of the country names match the colors in our UK and Ireland map and table at start of this document.

```r
plot(pca$x[,1], pca$x [,2],
     xlab="PC1", ylab= "PC2", xlim=c(-270,500),
     text(pca$x[,1], pca$x[,2], colnames(x), col=c("orange", "red", "blue", "darkgreen")))
```



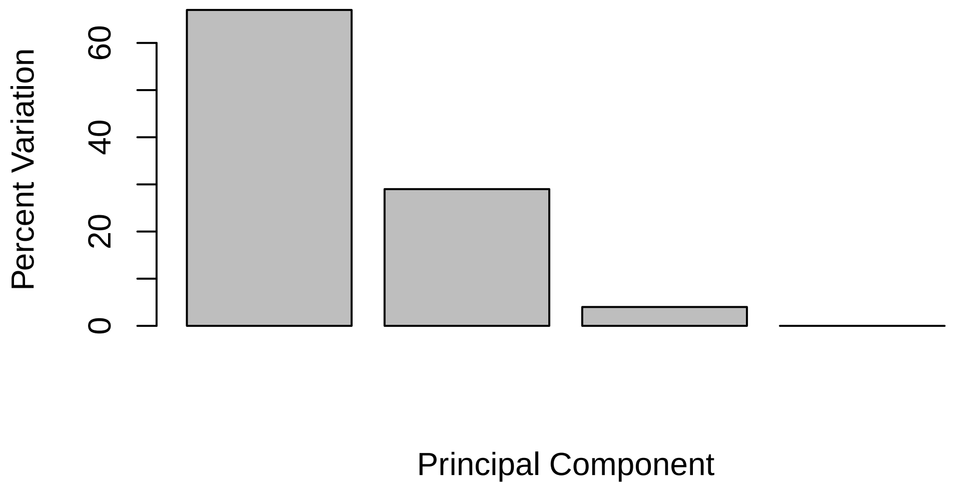
How much variation in the original data set does each PC account for (by proportion to 100)?

```r
v <- round (pca$sdev^2/sum(pca$sdev^2)*100)
v
```

```
[1] 67 29  4  0
```

Summarize as a barplot

```r
barplot(v, xlab="Principal Component", ylab="Percent Variation")
```



### Variable Loadings

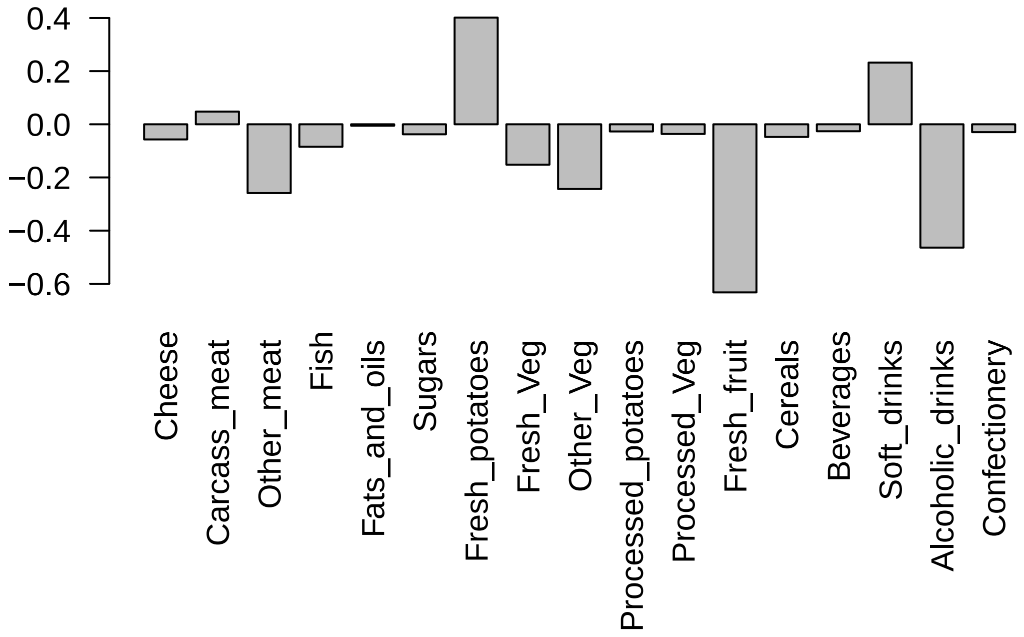
We can look at the so-caled PC loadings result to see how the original foods contribute to our new PCs (ie how the original variables contribute to our new, better variables).

```
pca$rotation[,1]
```

```
           Cheese        Carcass_meat          Other_meat                Fish 
     -0.056955380         0.047927628        -0.258916658        -0.084414983 
    Fats_and_oils              Sugars      Fresh_potatoes           Fresh_Veg 
     -0.005193623        -0.037620983         0.401402060        -0.151849942 
        Other_Veg  Processed_potatoes       Processed_Veg         Fresh_fruit 
     -0.243593729        -0.026886233        -0.036488269        -0.632640898 
          Cereals           Beverages         Soft_drinks     Alcoholic_drinks 
     -0.047702858        -0.026187756         0.232244140        -0.463968168 
    Confectionery 
     -0.029650201 
```

Positive values indicate more impact than others, negative values indicate less impact than others

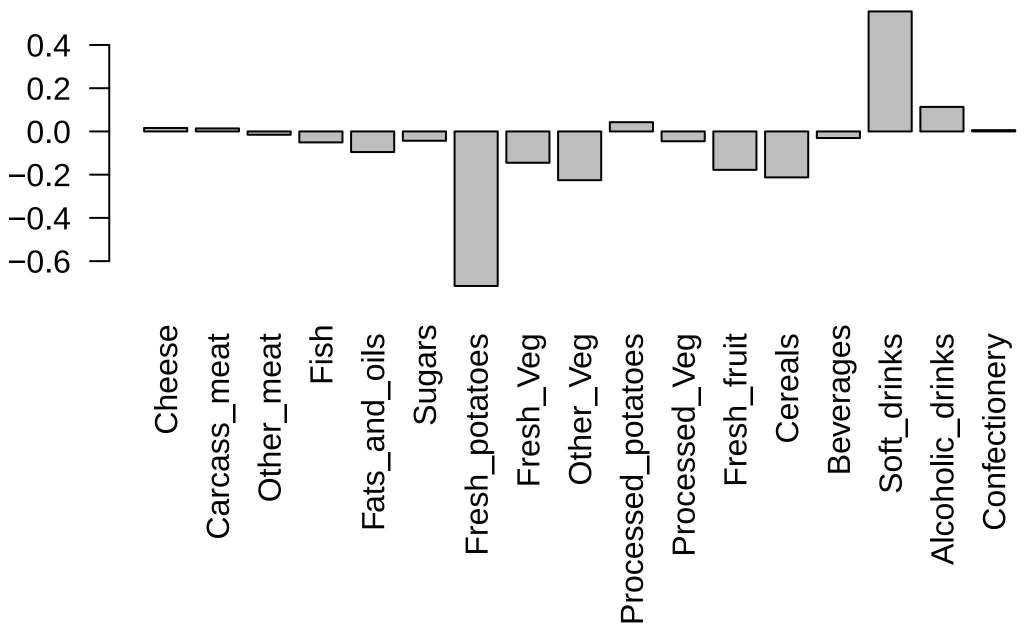
```
#Graphical focus on PC1, largest positive loading scores 'push' N. Ireland to the right posi
par(mar=c(10,3,0.35,0))
barplot(pca$rotation[,1], las=2)
```



> Q9: Generate a similar 'loadings plot' for PC2. What two food groups feature prominantely and what does PC2 maninly tell us about?

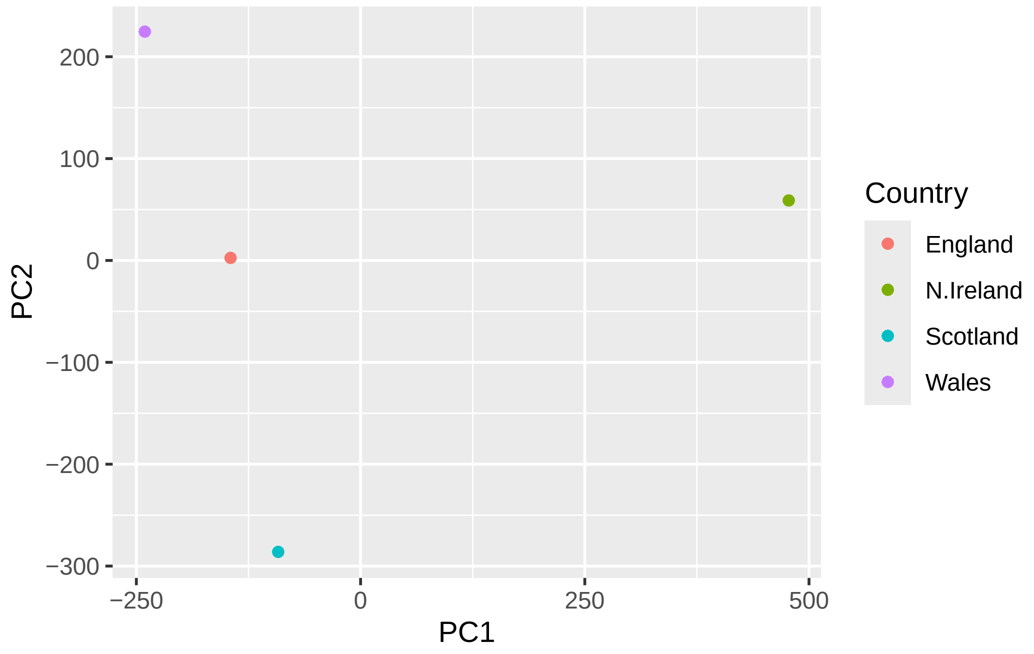
Fresh potatoes and soft drink are the most dominant bars in this graph. Soft drinks are significantly higher than average and fresh potatoes are significantly lower.

```
par(mar=c(10,3,0.35,0))
barplot(pca$rotation[,2], las=2)
```



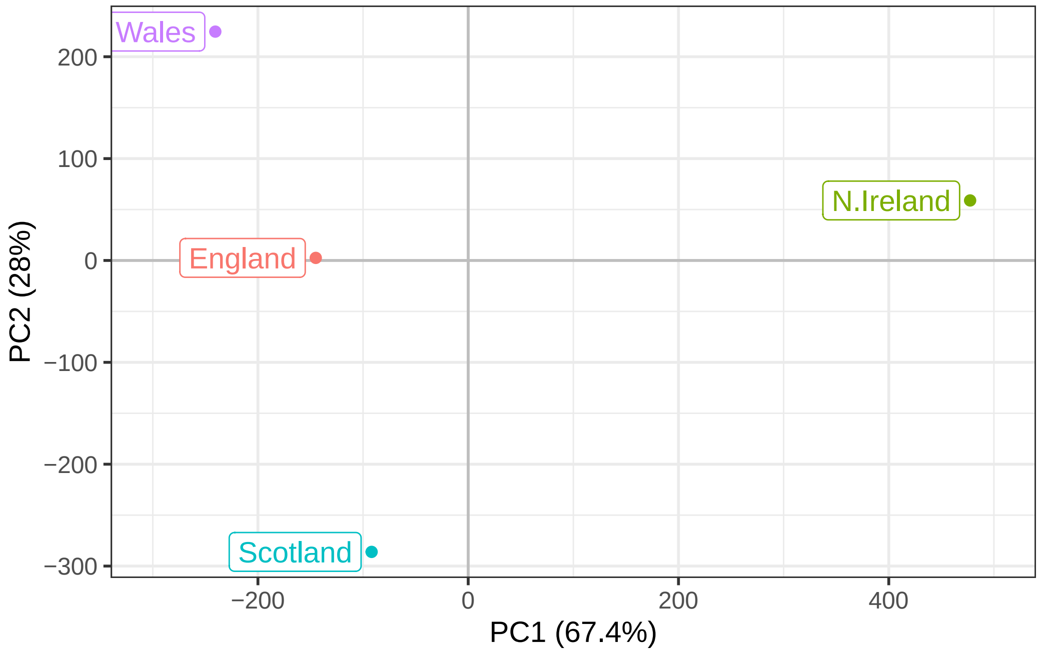
**Using ggplot for figures**

```
library(ggplot2)

df <- as.data.frame(pca$x)
df_lab <- tibble::rownames_to_column(df, "Country")

ggplot(df_lab) +
  aes(PC1, PC2, col=Country) +
  geom_point()
```



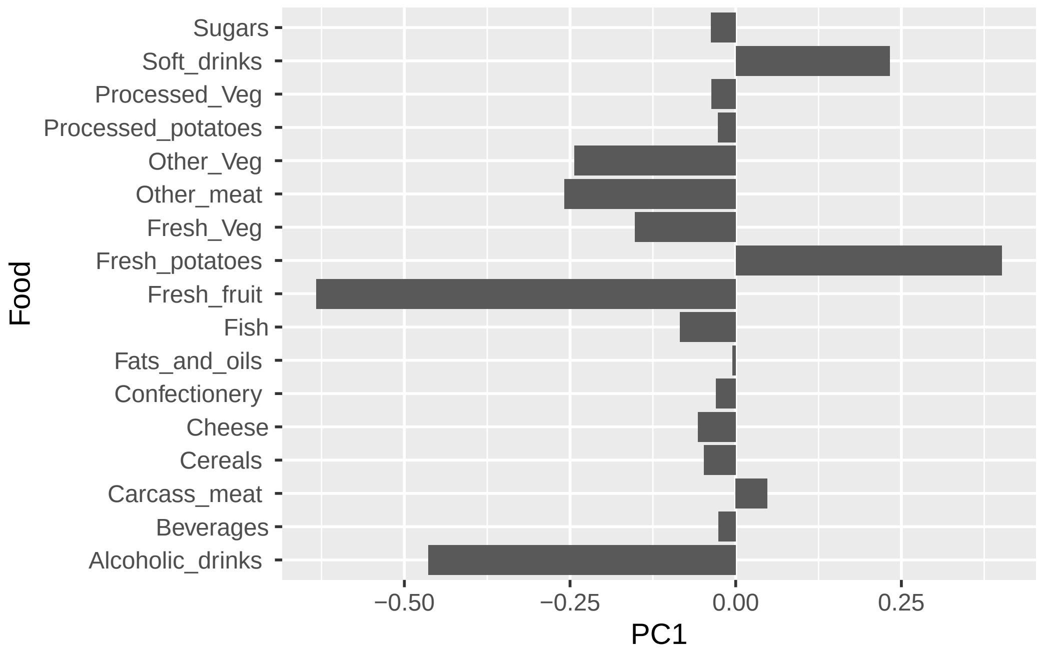
Fancier but a pain to make graph

```
ggplot(df_lab) +
  aes(PC1, PC2, col=Country, label=Country) +
  geom_hline(yintercept = 0, col="gray") +
  geom_vline(xintercept = 0, col="gray") +
  geom_point(show.legend = FALSE) +
  geom_label(hjust=1, nudge_x = -10, show.legend = FALSE) +
  expand_limits(x = c(-300,500)) +
  xlab("PC1 (67.4%)") +
  ylab("PC2 (28%)") +
  theme_bw()
```



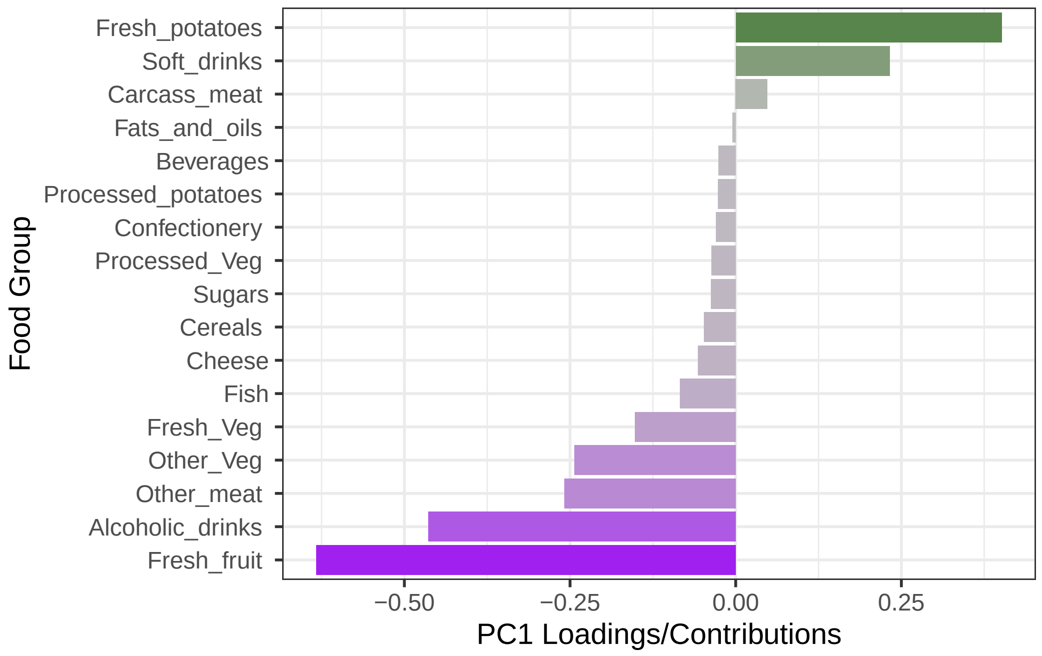
Plot of loadings and PC contributions

```r
ld <- as.data.frame(pca$rotation)
ld_lab <- tibble::rownames_to_column(ld, "Food")

ggplot(ld_lab) +
  aes(PC1, Food) +
  geom_col()
```



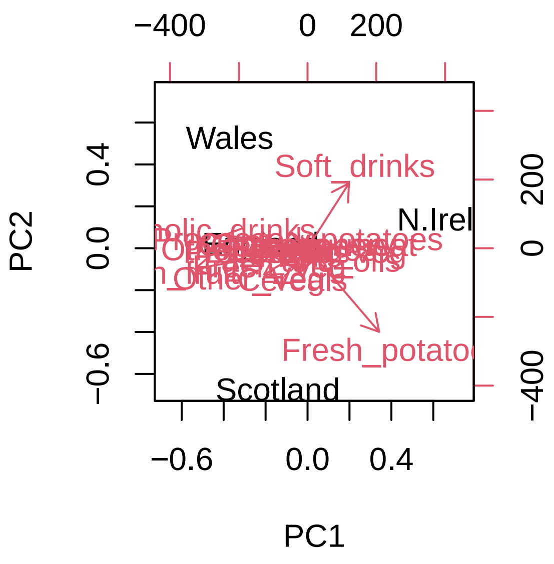
Prettier one

```
ggplot(ld_lab) +
  aes(PC1, reorder(Food, PC1), bg=PC1) +
  geom_col() +
  xlab("PC1 Loadings/Contributions") +
  ylab("Food Group") +
  scale_fill_gradient2(low="purple", mid="gray", high="darkgreen", guide=NULL) +
  theme_bw()
```



Biplots are a good option for small datasets. There is a central group of red aroows pointing to the red word labels for each variable.

```r
biplot(pca)
```



## PCA of RNA-seq data

```
url2 <- "https://tinyurl.com/expression-CSV"
rna.data <- read.csv(url2, row.names=1)
head(rna.data)
```

```
       wt1 wt2  wt3  wt4 wt5 ko1 ko2 ko3 ko4 ko5
gene1  439 458  408  429 420  90  88  86  90  93
gene2  219 200  204  210 187 427 423 434 433 426
gene3 1006 989 1030 1017 973 252 237 238 226 210
gene4  783 792  829  856 760 849 856 835 885 894
gene5  181 249  204  244 225 277 305 272 270 279
gene6  460 502  491  491 493 612 594 577 618 638
```

Q10: How many genes and samples are in this data set?

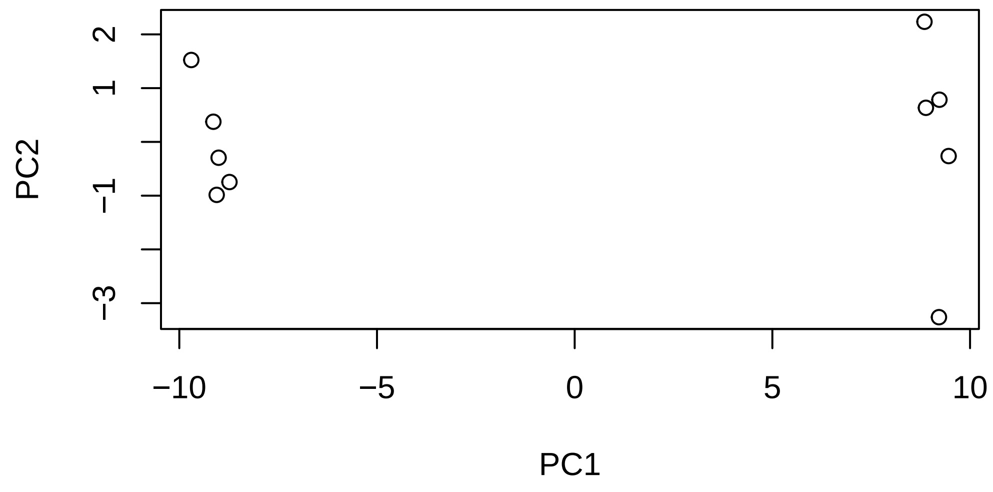
10 genes, 100 samples

```
dim(rna.data)
```

```
[1] 100  10
```

Run a PCA and plot the results

```
pca <- prcomp(t(rna.data), scale=TRUE)
#what does the t indicate here?
plot(pca$x[,1], pca$x[,2], xlab="PC1", ylab="PC2")
```

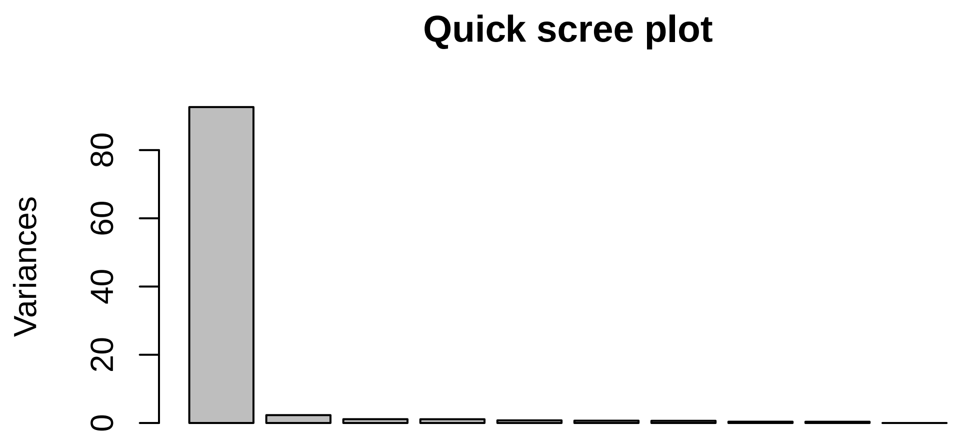


```r
summary(pca)
```

```
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7
Standard deviation     9.6237 1.5198 1.05787 1.05203 0.88062 0.82545 0.80111
Proportion of Variance 0.9262 0.0231 0.01119 0.01107 0.00775 0.00681 0.00642
Cumulative Proportion  0.9262 0.9493 0.96045 0.97152 0.97928 0.98609 0.99251
                           PC8     PC9      PC10
Standard deviation     0.62065 0.60342 3.345e-15
Proportion of Variance 0.00385 0.00364 0.000e+00
Cumulative Proportion  0.99636 1.00000 1.000e+00
```

Notice 92% of variance is contained in PC1.

```r
plot(pca, main="Quick scree plot")
```

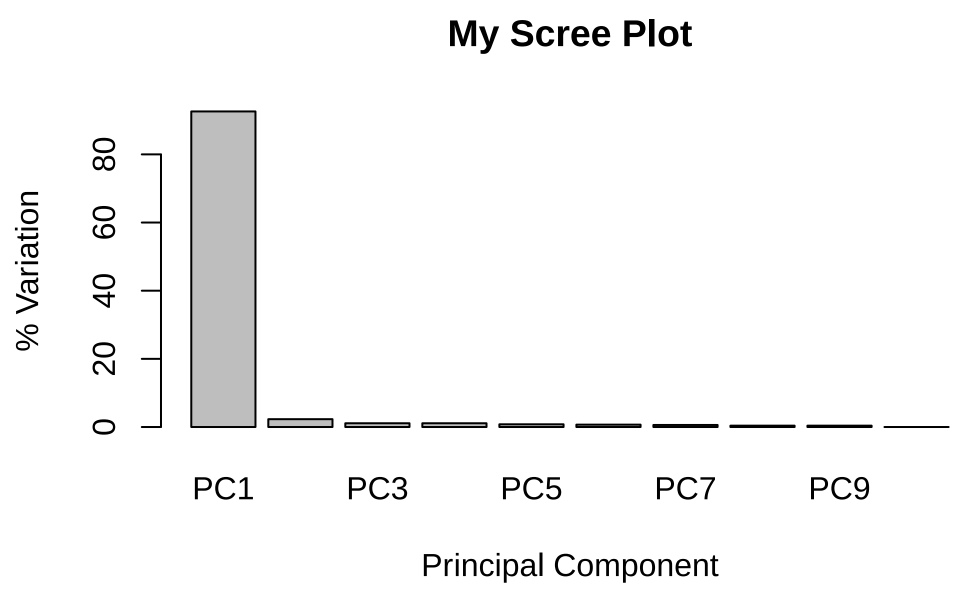


Let's make the scree plot on our own.

```
# Variance captured per PC
pca.var <- pca$sdev^2

#percent variance
pca.var.per <- round(pca.var/sum(pca.var)*100, 1)
pca.var.per
```

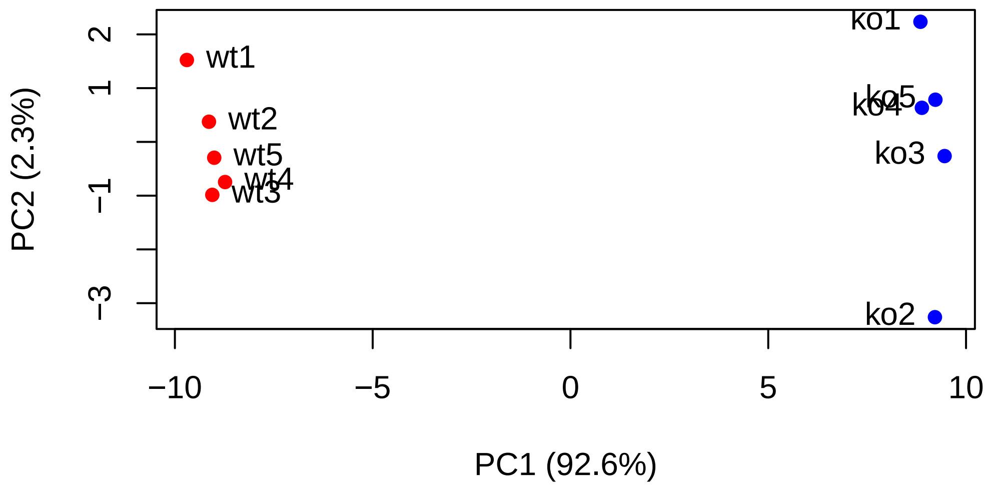
```
 [1] 92.6  2.3  1.1  1.1  0.8  0.7  0.6  0.4  0.4  0.0
```

```
barplot(pca.var.per, main= "My Scree Plot",
        names.arg= paste0("PC", 1:10),
        xlab="Principal Component", ylab="% Variation")
```



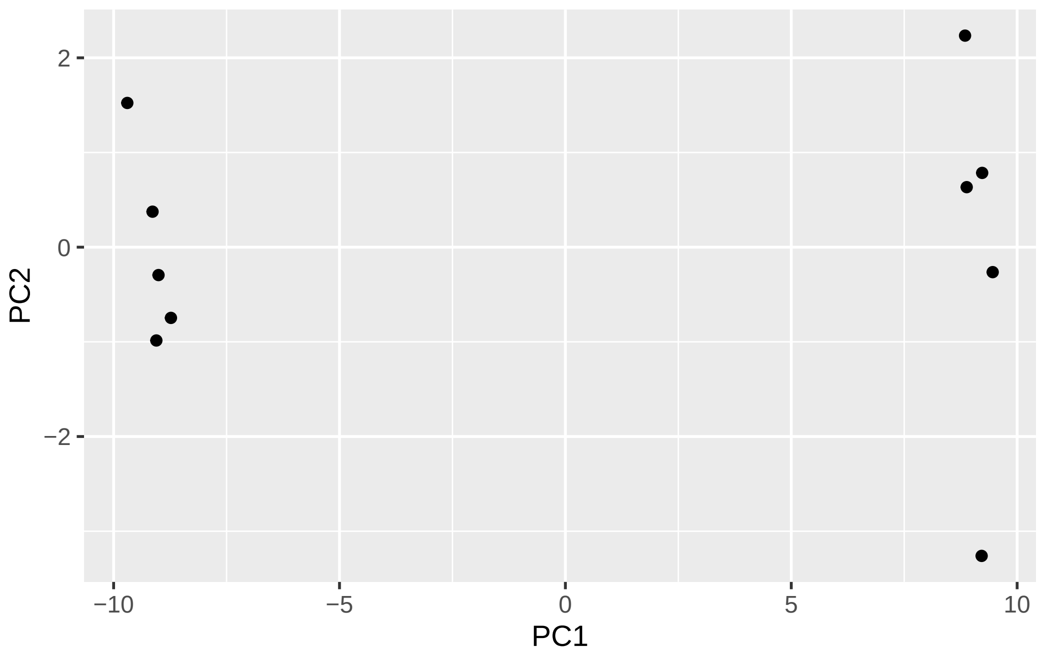
More attractive and more useful, labeling specific points by name and with colors

```r
colvec <- colnames(rna.data)
colvec[grep("wt", colvec)] <- "red"
colvec[grep("ko", colvec)] <- "blue"

plot(pca$x[,1], pca$x[,2], col=colvec, pch=16,
     xlab=paste0("PC1 (", pca.var.per[1], "%)"),
     ylab=paste0("PC2 (", pca.var.per[2], "%)"))

text(pca$x[,1], pca$x[,2], labels = colnames(rna.data), pos=c(rep(4,5), rep(2,5)))
```



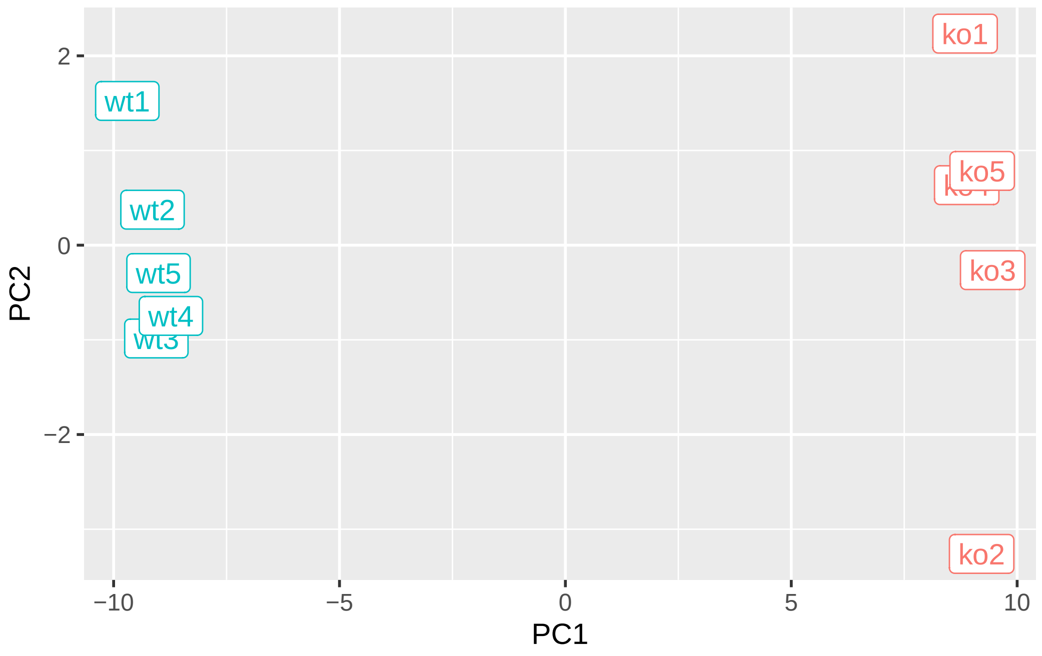
Let's graph it by ggplot

```
library(ggplot2)

df <- as.data.frame(pca$x)

ggplot(df) +
  aes(PC1, PC2) +
  geom_point()
```



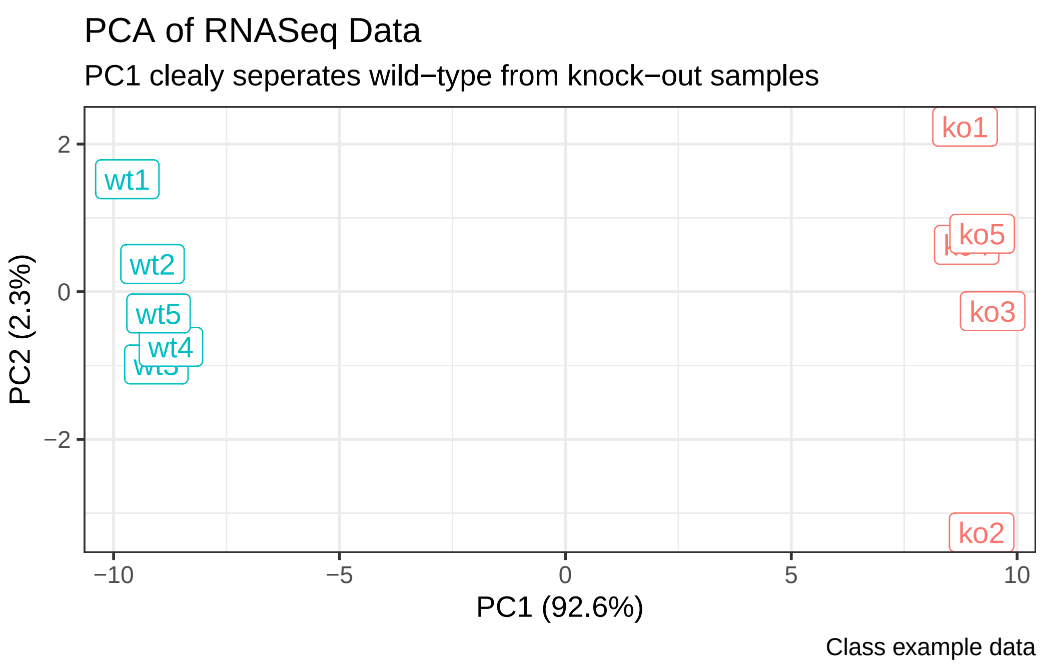
Add a condition specific color, sample label aesthetic for WT vs knockout

```
df$samples <- colnames(rna.data)
df$condition <- substr(colnames(rna.data),1,2)

p <- ggplot(df) +
        aes(PC1, PC2, label=samples, col=condition) +
        geom_label(show.legend = FALSE)
p
```



Add title, caption, x and y specificity, and a theme

```
p + labs(title="PCA of RNASeq Data",
       subtitle = "PC1 clealy seperates wild-type from knock-out samples",
       x=paste0("PC1 (", pca.var.per[1], "%)"),
       y=paste0("PC2 (", pca.var.per[2], "%)"),
       caption="Class example data") +
     theme_bw()
```


PCA of RNASeq Data
PC1 clealy seperates wild−type from knock−out samples
wt1
wt2
wt5
wt4
wt3
ko1
ko5
ko4
ko3
ko2
PC1 (92.6%)
PC2 (2.3%)
−10
−5
0
5
10
2
0
−2
Class example data