Lab 10. Dimensionality reduction. PCA

```r
library(tidyverse)
library(ggfortify)
#Sys.setlocale(locale = "ru_RU.UTF-8")
```

Principal component analysis (PCA)

1. Main problem

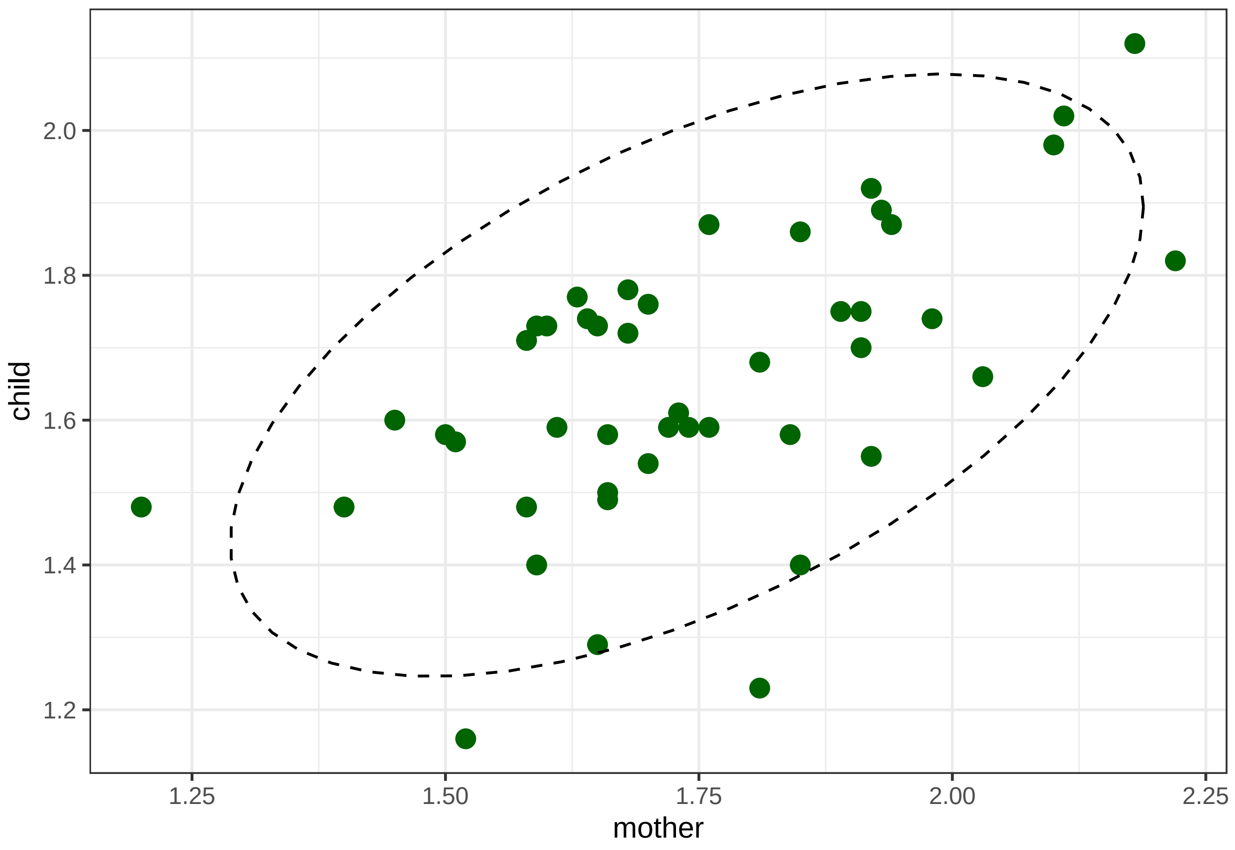
Sometimes you have a huge amount of variables. So, to make your data profitable you need to reduce number of variables saving without losing the precious information.

- Principal component analysis (PCA)
- Linear discriminant analysis (LDA)
- Multidimensional scaling (MDS)
- ...

2. Data

This is a dataset from [Huttenlocher, Vasilyeva, Cymerman, Levine 2002]. The authors analysed 46 pairs of mothers and children (aged from 47 to 59 months, mean age – 54). They recorded and transcribed 2 hours from each child per day. In their study, they compared the number of noun phrases per utterance in mother speech to the number of noun phrases per utterance in child speech.

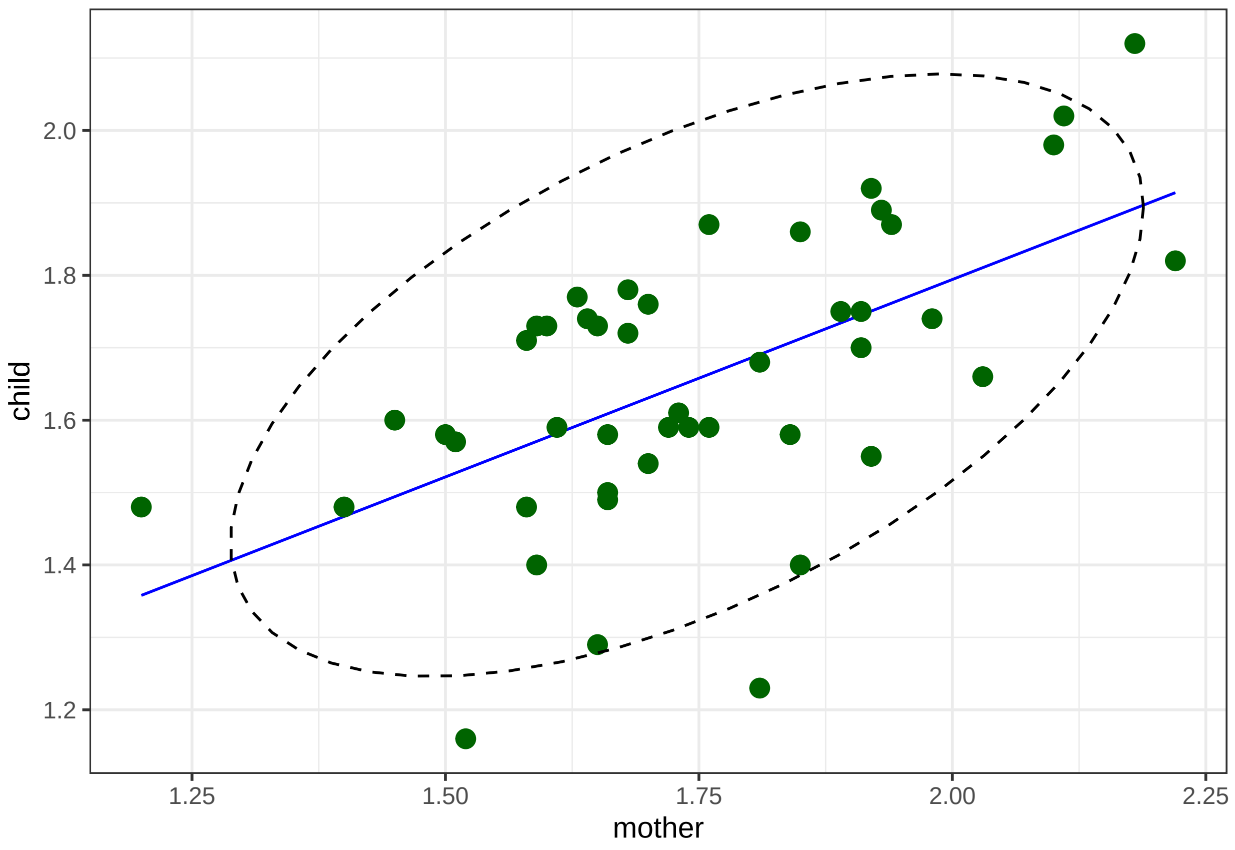
```r
df <- read_csv("https://raw.githubusercontent.com/LingData2019/LingData2020/master/data/Huttenlocher.cs

df %>%
  ggplot(aes(mother, child))+
  geom_point(color = "darkgreen", size = 3)+
  stat_ellipse(linetype=2)+
  theme_bw()
```



3. PCA

PCA is essentially a rotation of the coordinate axes, chosen such that each successful axis captures as much variance as possible. We can reduce 2 dementions to one using a regression:

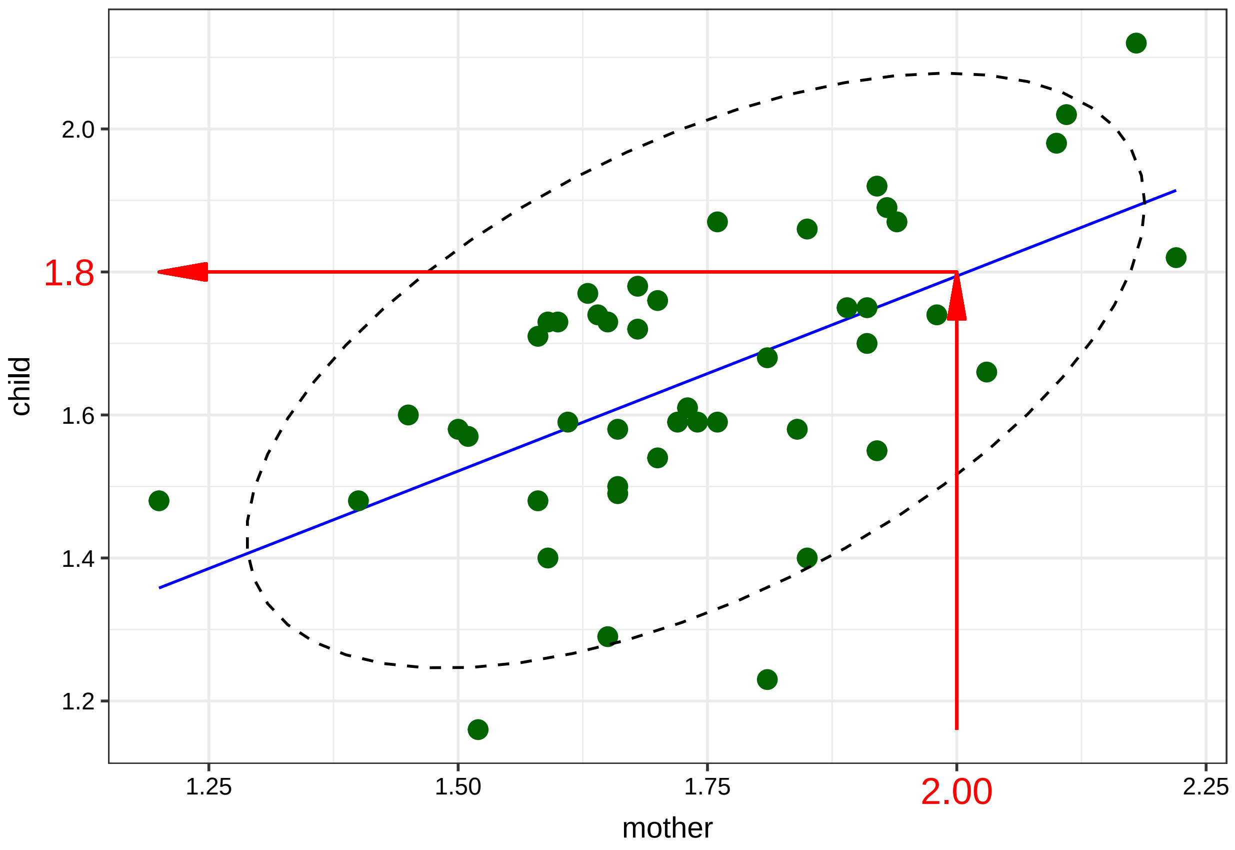
```
fit <- lm(child~mother, data = df)
df$model <- predict(fit)

p1 <- df %>%
  ggplot(aes(mother, child)) +
  geom_line(aes(mother, model), color = "blue") +
  geom_point(color = "darkgreen", size = 3) +
  stat_ellipse(linetype=2) +
  scale_y_continuous(breaks = c(1.2, 1.4, 1.6, 1.8, 2.0)) +
  theme_bw()
p1
```



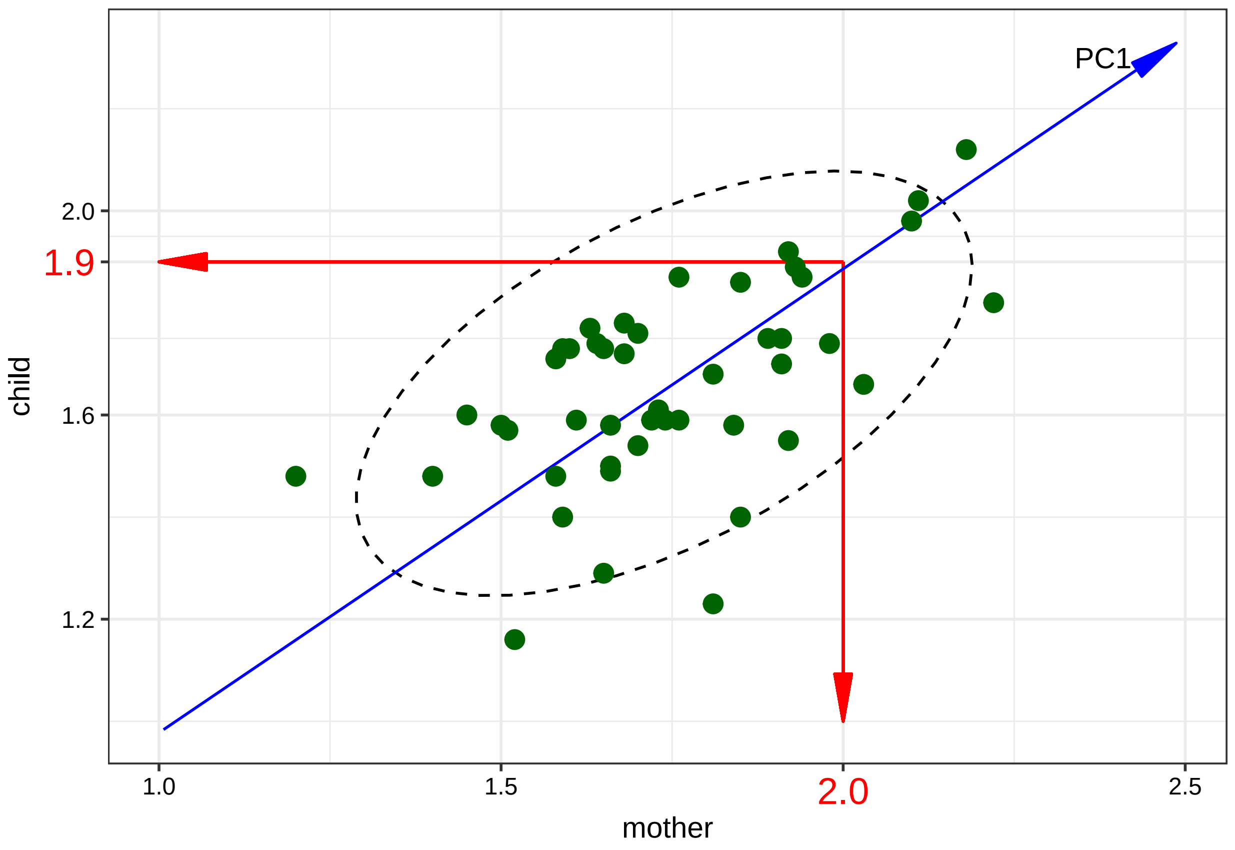
We used regression for predicting value of one variable by another variable.

```
p1 +
# plot red arrows
  geom_segment(aes(x=min(mother), y=1.8, xend=2, yend=1.8), size=0.5, color = "red",
               arrow =  arrow(angle = 10, type = "closed", ends = "first")) +
  geom_segment(aes(x=2, y=min(child), xend=2, yend=1.8), size=0.5, color = "red",
               arrow =  arrow(angle = 10, type = "closed")) +
# pin two points on axes
  theme(axis.text.x = element_text(color=c("black", "black", "black", "red", "black"), size=c(9, 9, 9, 
      axis.text.y = element_text(color=c("black", "black", "black", "red", "black", "black"), size=c(9,
```

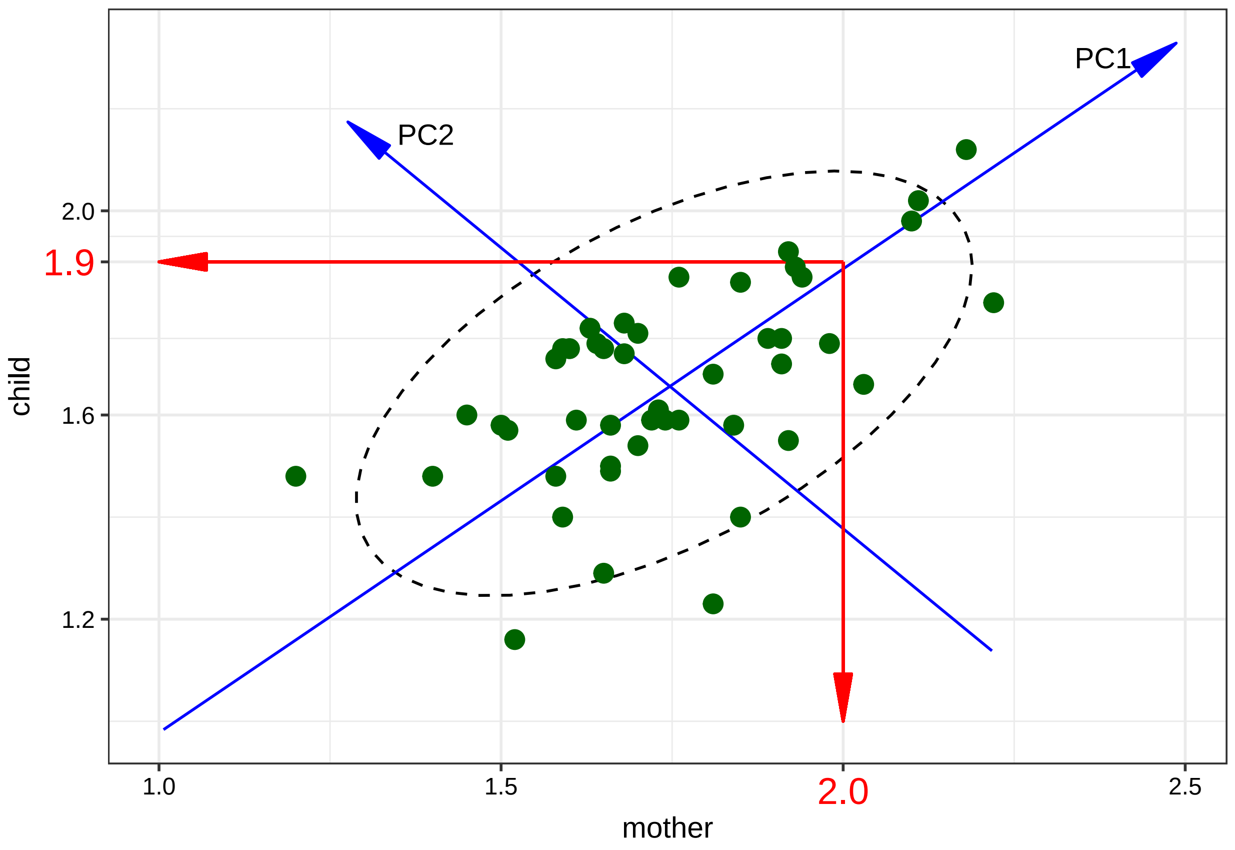
```
## Warning: Vectorized input to `element_text()` is not officially supported.
## Results may be unexpected or may change in future versions of ggplot2.

## Warning: Vectorized input to `element_text()` is not officially supported.
## Results may be unexpected or may change in future versions of ggplot2.
```



In PCA we change coordinate system and start predicting variables' values using less variables.



The blue line is *the first Princple Component* (and it is NOT a regression line). The number of the PCs is always equal to the number of variables. So we can draw the second PC:



The main point of PCA is that if cumulative proportion of explained variance is high we can drop some PCs. So, we need know the following things:

- What is the cumulative proportion of explained variance?

```
df <- read_csv("https://raw.githubusercontent.com/LingData2019/LingData2020/master/data/Huttenlocher.cs
summary(prcomp(df))
```

```
## Importance of components:
##                           PC1    PC2
## Standard deviation     0.2544 0.1316
## Proportion of Variance 0.7890 0.2110
## Cumulative Proportion  0.7890 1.0000
```

We see that PC1 explains only 78.9% of the variance in our data.

- How PCs are rotated comparing to the old axes?

```
prcomp(df)
```

```
## Standard deviations (1, .., p=2):
## [1] 0.2543899 0.1315688
##
## Rotation (n x k) = (2 x 2):
##              PC1        PC2
## child  0.6724959 -0.7401009
## mother 0.7401009  0.6724959
```

So the formula for the first component rotation is

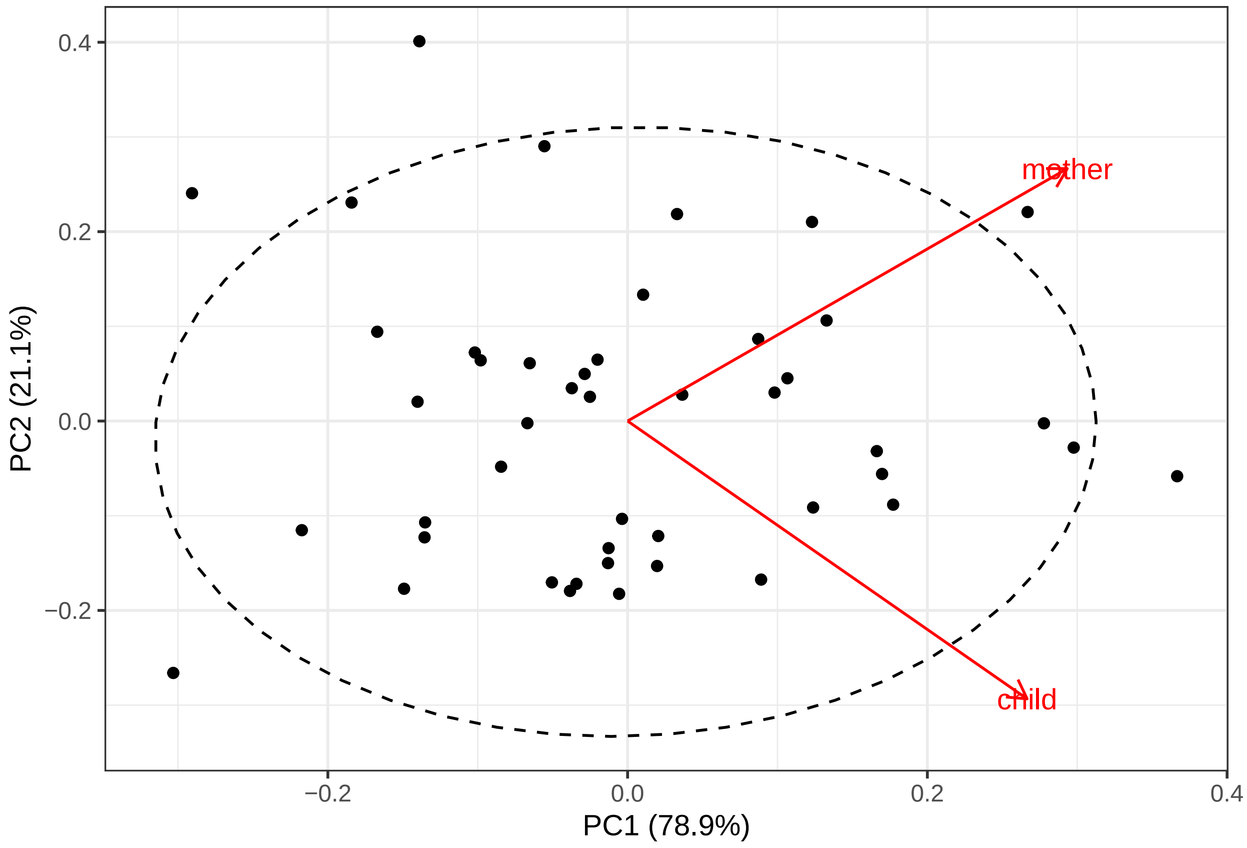
$$PC1 = 0.6724959 \times child + 0.7401009 \times mother$$

The formula for the second component rotation is

$$PC2 = -0.7401009 \times child + 0.6724959 \times mother$$

Now we can change the axes. We use the `autoplot()` function from `ggfortify` package to produce the graph:

```r
autoplot(pca,
         loadings = TRUE,
         loadings.label = TRUE) +
  theme_bw() +
  stat_ellipse(linetype=2)
```



Summary:

- If the cumulative proportion of explained variance for some PCs is high, we can change coordinate system and start predicting variables' values using less variables.
- We can even make a regresion or clusterisation model.
- PCA for categorical variables is called Multiple correspondence analysis (MCA)

R functions

There are several functions for PCA, MCA and their visualisation.

- PCA: prcomp()
- PCA: princomp()
- PCA: FactoMineR::PCA()

- PCA: ade4::dudi.pca()
- PCA: amap::acp()
- PCA visualisation: ggfortify::autoplot

2 Gospels' frequency word lists

The gospels of Matthew, Mark, and Luke are referred to as the Synoptic Gospels and stand in contrast to John, whose content is comparatively distinct. This dataset (https://tinyurl.com/y8tcf3uw) contains frequency of selected words (without stopwords, without pronouns and without frequent word "Jesus") as attested in four gospels of the New Testament.

For some visualisations you will need assign row names to the dataframe:

```r
gospels <- read.csv("https://tinyurl.com/y8tcf3uw")
row.names(gospels) <- gospels$word
```

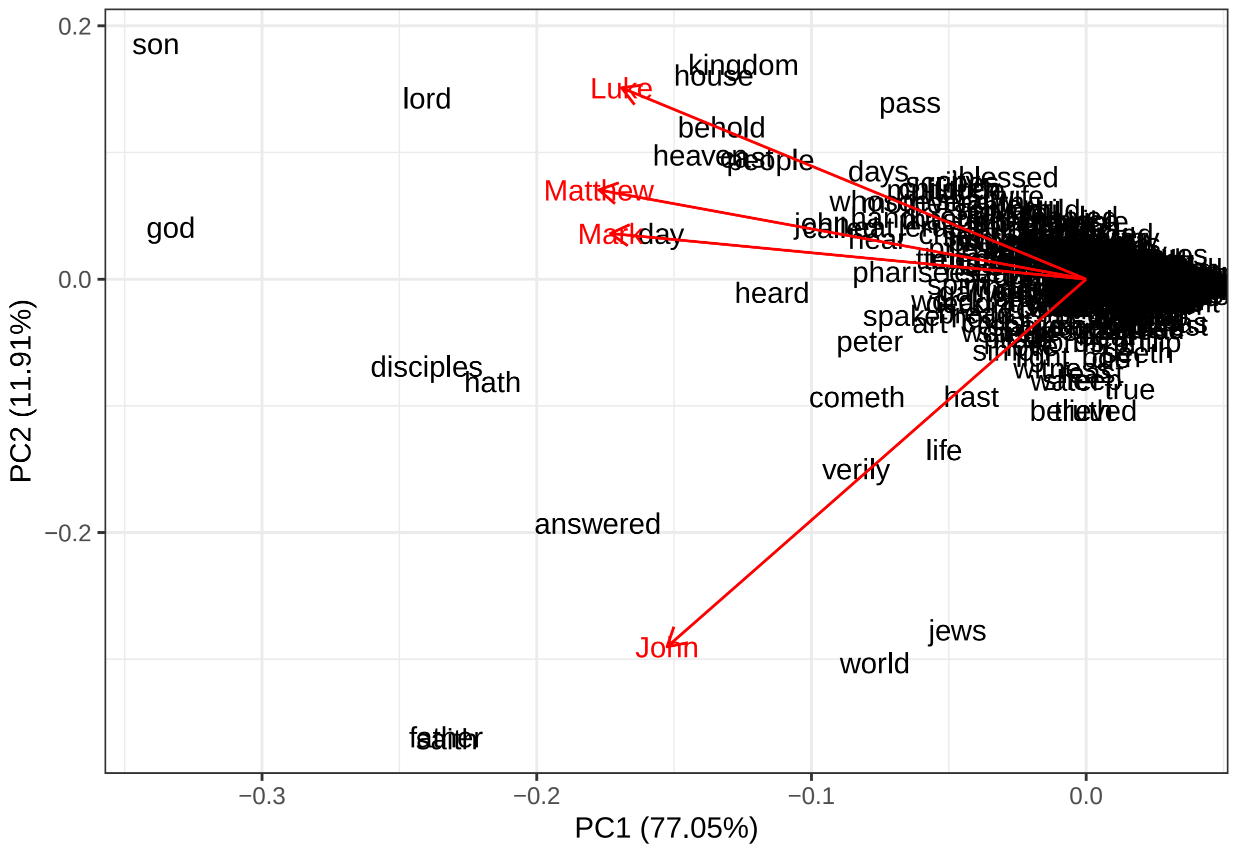
```r
PCA <- prcomp(gospels[,2:5], center = TRUE, scale. = TRUE)
summary(PCA)
```

1.2 Apply PCA to four continuous variables. Use `prcomp()` function. What is the cumulative proportion of explained variance for the first and second component?

```
## Importance of components:
##                           PC1    PC2     PC3     PC4
## Standard deviation     1.7556 0.6903 0.50983 0.42619
## Proportion of Variance 0.7705 0.1191 0.06498 0.04541
## Cumulative Proportion  0.7705 0.8896 0.95459 1.00000
```

2.2 Use the `autoplot()` function of the library ggfortify for creating plot like this. See more examples here: https://cran.r-project.org/web/packages/ggfortify/vignettes/plot_pca.html

```r
autoplot(PCA,
         shape = FALSE,
         loadings = TRUE,
         label = TRUE,
         loadings.label = TRUE)+
  theme_bw()
```



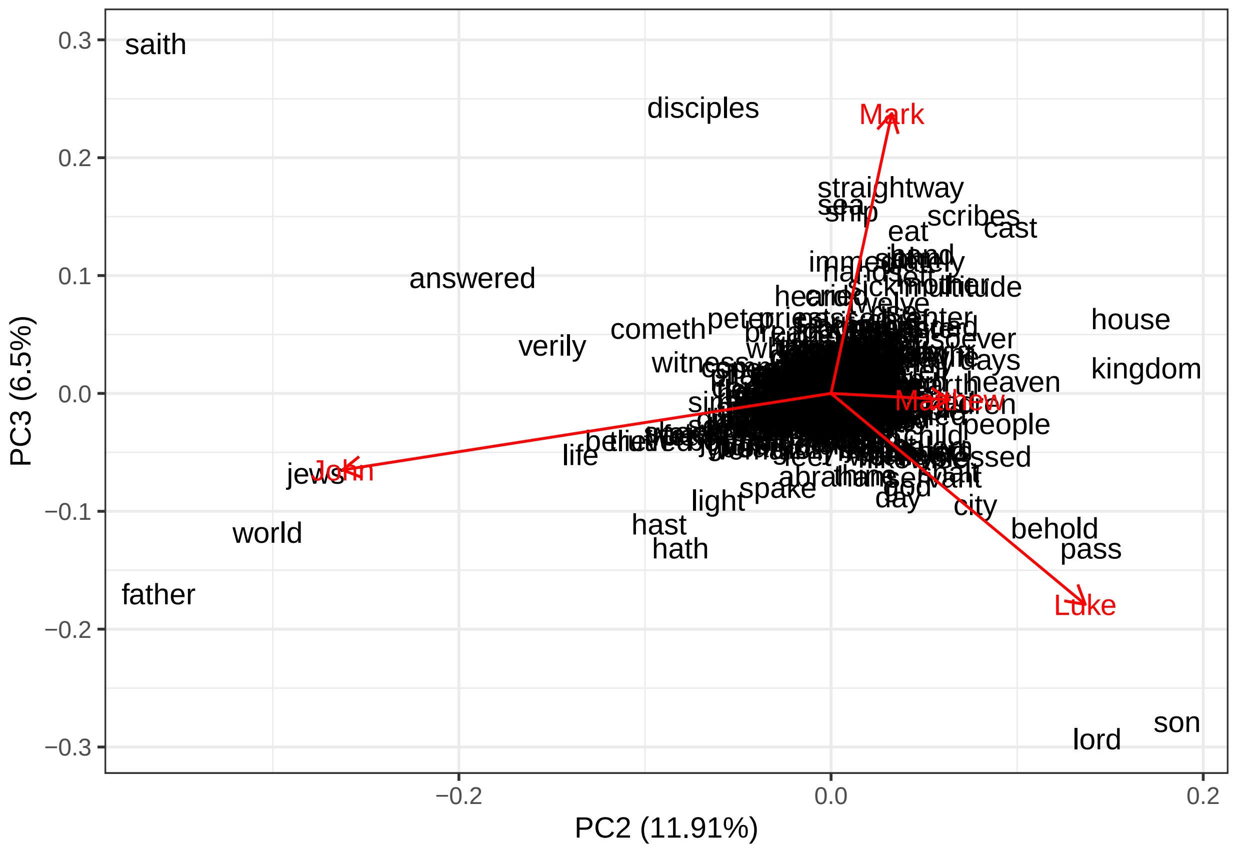
```r
predict(PCA, data.frame(John = 0.05, Luke = 0.01, Mark = 0.02, Matthew = 0.02))
```

2.3 Predict the coordinates for the word "Jesus", which have the following frequencies: John = 0.05, Luke = 0.01, Mark = 0.02, Matthew = 0.02.

```
##             PC1       PC2      PC3      PC4
## [1,] -22.60497 -9.599171 2.367918 2.104944
```

We can also look at PC2 and PC3 components:

```r
autoplot(PCA, x=2, y=3,
         shape = FALSE,
         loadings = TRUE,
         label = TRUE,
         loadings.label = TRUE)+
  theme_bw()
```



Useful links

- FactoMineR for PCA link