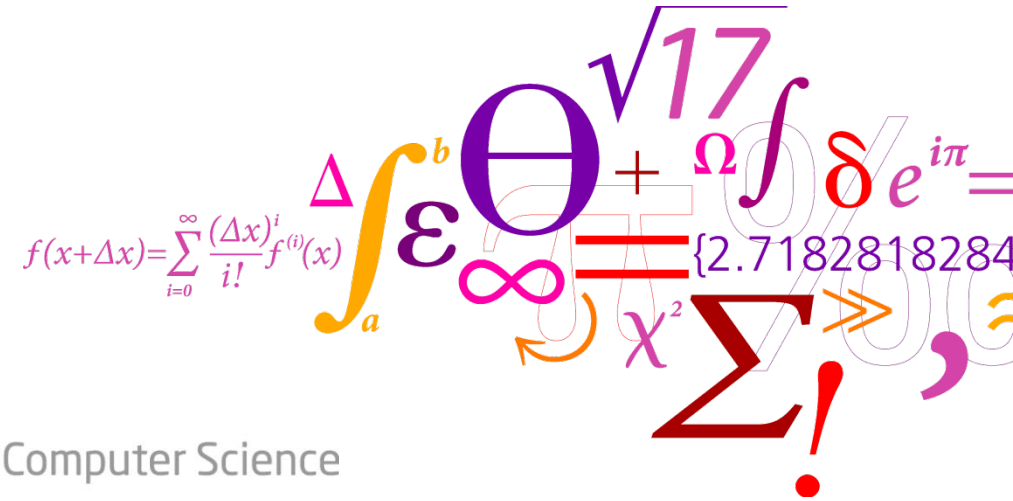
Principal Components Analysis

January 10th, 2025

Anders Stockmarr
Section for Statistics and Data Analysis, DTU
anst@dtu.dk



DTU Compute
Department of Applied Mathematics and Computer Science

Programme

- Monday: Statistical Inference, the t-test
- Tuesday: Simple and Multiple Regression
- Wednesday: ANOVA, ANCOVA, and Linear Models
- Thursday: Categorical Data, Writing Statistical Reports, Logistic regression
- Friday: Repeated Measurements, **Principal Component Analysis**

Contents:

Introduction

PCA: Modeling Data

- In all previous parts of the course, data has been subdivided into two classes:
  - Response variables;
  - explanatory variables.
- The data were analyzed in order to identify impact of the explanatory variable on the response variables.
- What if the target of analysis isn't to identify effects;
- But to **uncover the structure** of a complicated set of data?

This is the essence of the use of

Principal Components Analysis - PCA

Main example: The Wine data

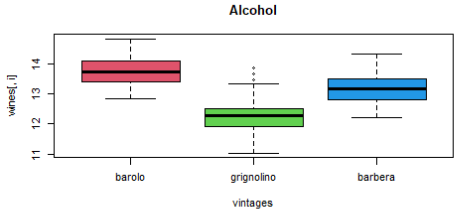
```
load("Data/Winedata.Rdata")
```

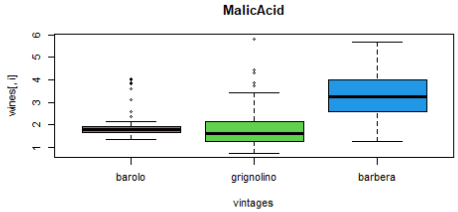
- A dataset consisting of n=178 Italian wines. Of these, 59 are Barolo wines, 71 are Grignolino wines, and 48 are Barbera wines. Two sub-elements: `wines` and `vintages` (Barolo, Grignolino or Barbera).

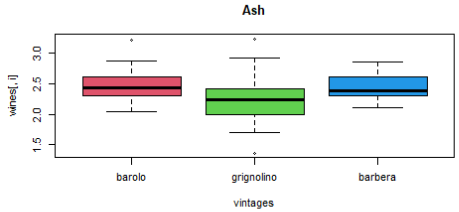
- 13 characteristics of the wines:

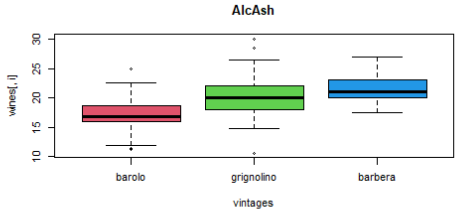
1) Alcohol
2) MalicAcid
3) Ash
4) Alkalinity of ash: AlcAsh
5) Magnesium: Mg
6) Total Phenols: Phenols
7) Flavanoids: Flav
8) Nonflavanoid phenols: NonFlavPhenols
9) Proanthocyanins: Proa
10) Color intensity: Color
11) Color hue: Hue
12) OD280/OD315 protein measurement: OD
13) Proline (amino acid): Proline

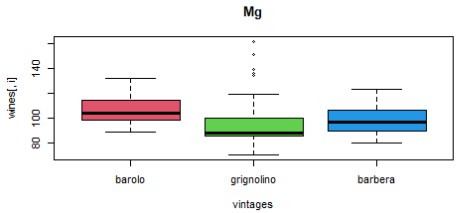
Variation of Vintages

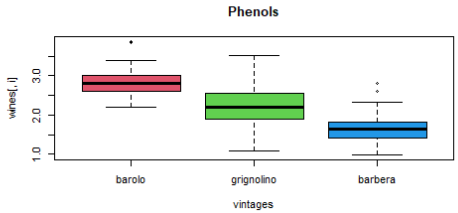


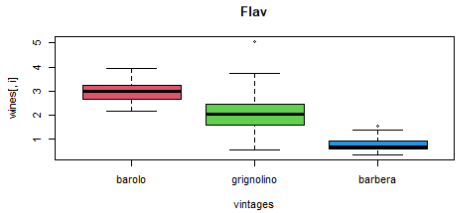


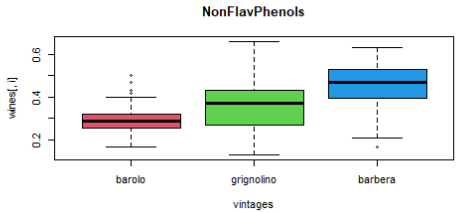


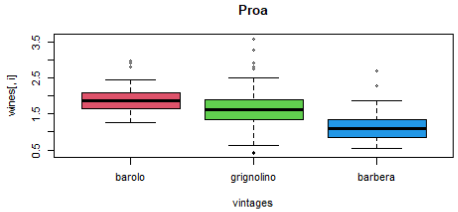


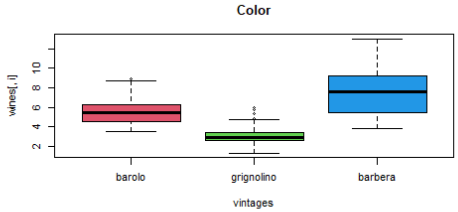


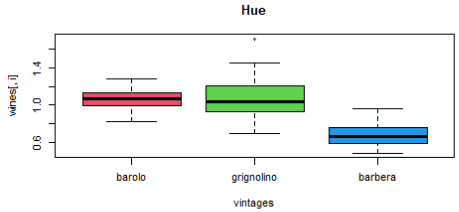


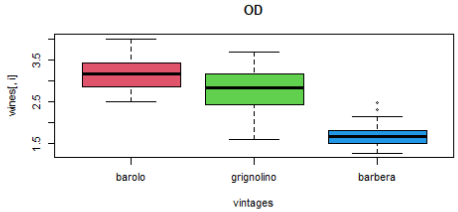


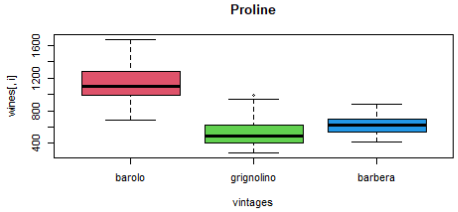




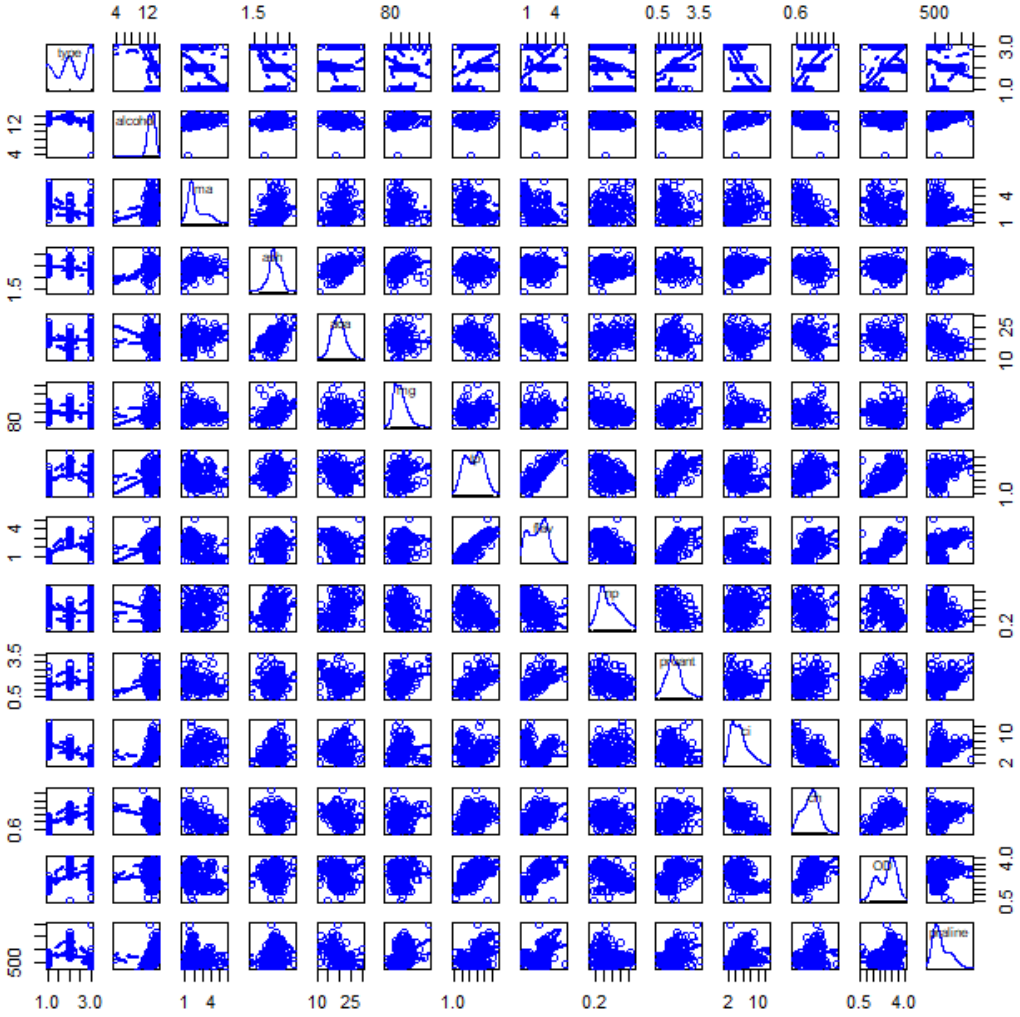








The Wine Data



Separation of Wines

- What causes wines to be different?
- With the 13 characteristics, we can distinguish wines through differences in the charateristics. But are all 13 characteristics necessary? Some may be redundant.
- If we can identify scales (linear combinations of the characteristics) where the characteristics vary the most, we can also find a scale that differentiates optimally between the wines.

Separation of Wines

- The variance covariance matrix of the wine characteristics:

```
round(var(wines),digits=2)
              Alcohol MalicAcid   Ash  AlcAsh      Mg Phenols  Flav NonFlavPhenols  Proa  Color   Hue    OD  Proline
Alcohol          0.66      0.09  0.05   -0.84    3.14    0.15  0.19          -0.02  0.06   1.03 -0.01  0.04   164.57
MalicAcid        0.09      1.25  0.05    1.08   -0.87   -0.23 -0.46           0.04 -0.14   0.64 -0.14 -0.29   -67.55
Ash              0.05      0.05  0.08    0.41    1.12    0.02  0.03           0.01  0.00   0.16  0.00  0.00    19.32
AlcAsh          -0.84      1.08  0.41   11.15   -3.97   -0.67 -1.17           0.15 -0.38   0.15 -0.21 -0.66  -463.36
Mg               3.14     -0.87  1.12   -3.97  203.99    1.92  2.79          -0.46  1.93   6.62  0.18  0.67  1769.16
Phenols          0.15     -0.23  0.02   -0.67    1.92    0.39  0.54          -0.04  0.22  -0.08  0.06  0.31    98.17
Flav             0.19     -0.46  0.03   -1.17    2.79    0.54  1.00          -0.07  0.37  -0.40  0.12  0.56   155.45
NonFlavPhenols  -0.02      0.04  0.01    0.15   -0.46   -0.04 -0.07           0.02 -0.03   0.04 -0.01 -0.04   -12.20
Proa             0.06     -0.14  0.00   -0.38    1.93    0.22  0.37          -0.03  0.33  -0.03  0.04  0.21    59.55
Color            1.03      0.64  0.16    0.15    6.62   -0.08 -0.40           0.04 -0.03   5.37 -0.28 -0.71   230.77
Hue             -0.01     -0.14  0.00   -0.21    0.18    0.06  0.12          -0.01  0.04  -0.28  0.05  0.09    17.00
OD               0.04     -0.29  0.00   -0.66    0.67    0.31  0.56          -0.04  0.21  -0.71  0.09  0.50    69.93
Proline        164.57    -67.55 19.32 -463.36 1769.16   98.17 155.45         -12.20 59.55 230.77 17.00 69.93 99166.72
```

Separation of Wines

- To avoid scaling problems, we must scale the data to the same scale.
- The `scale` function in R subtracts the mean and divide by the sd:

$$X^{scaled} = \frac{X - mean(X)}{sd(X)}$$

```
round(var(scale(wines)),digits=2)
```

	Alcohol	MalicAcid	Ash	AlcAsh	Mg	Phenols	Flav	NonFlavPhenols	Proa	Color	Hue	OD	Proline
Alcohol	**1.00**	0.09	0.21	-0.31	0.27	0.29	0.24	-0.16	0.14	0.55	-0.07	0.07	0.64
MalicAcid	0.09	**1.00**	0.16	0.29	-0.05	-0.34	-0.41	0.29	-0.22	0.25	-0.56	-0.37	-0.19
Ash	0.21	0.16	**1.00**	0.44	0.29	0.13	0.12	0.19	0.01	0.26	-0.07	0.00	0.22
AlcAsh	-0.31	0.29	0.44	**1.00**	-0.08	-0.32	-0.35	0.36	-0.20	0.02	-0.27	-0.28	-0.44
Mg	0.27	-0.05	0.29	-0.08	**1.00**	0.21	0.20	-0.26	0.24	0.20	0.06	0.07	0.39
Phenols	0.29	-0.34	0.13	-0.32	0.21	**1.00**	0.86	-0.45	0.61	-0.06	0.43	0.70	0.50
Flav	0.24	-0.41	0.12	-0.35	0.20	0.86	**1.00**	-0.54	0.65	-0.17	0.54	0.79	0.49
NonFlavPhenols	-0.16	0.29	0.19	0.36	-0.26	-0.45	-0.54	**1.00**	-0.37	0.14	-0.26	-0.50	-0.31
Proa	0.14	-0.22	0.01	-0.20	0.24	0.61	0.65	-0.37	**1.00**	-0.03	0.30	0.52	0.33
Color	0.55	0.25	0.26	0.02	0.20	-0.06	-0.17	0.14	-0.03	**1.00**	-0.52	-0.43	0.32
Hue	-0.07	-0.56	-0.07	-0.27	0.06	0.43	0.54	-0.26	0.30	-0.52	**1.00**	0.57	0.24
OD	0.07	-0.37	0.00	-0.28	0.07	0.70	0.79	-0.50	0.52	-0.43	0.57	**1.00**	0.31
Proline	0.64	-0.19	0.22	-0.44	0.39	0.50	0.49	-0.31	0.33	0.32	0.24	0.31	**1.00**

Separation of Wines

- Let us consider the correlation matrix:

```
X<-var(scale(wines))
```

The sum of the standardized variances:

```
sum(diag(X))
[1] 13
```

Of course – there are 13 variables.

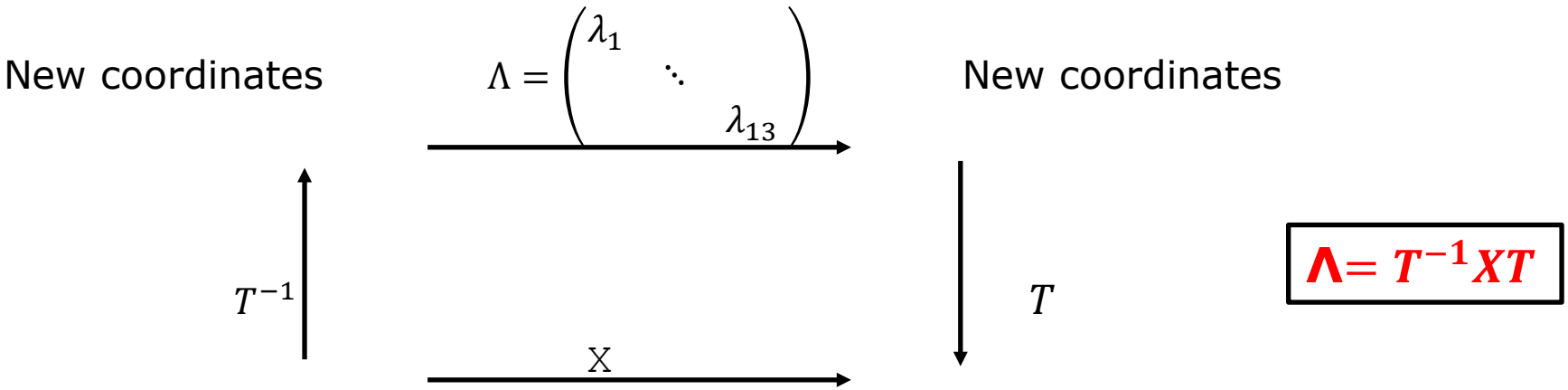
Original question: In which direction (scale) do wines data vary the most?

Let us start by representing the data in a set of coordinates where no correlation is present, to get an overview not disturbed by correlations.

Separation of Wines

- No correlations means that X is represented by a diagonal matrix in these directions;
- in other words that the new coordinates T consists of eigenvectors for X; solution to the equation

$$Xv = \lambda v$$



Separation of Wines

- Eigenvectors in R:

```
T<-eigen(X)$vectors
```

The inverse of T is equal to the matrix transpose t(T):

```
Lambda<-t(T)%*%X%*%T
round(Lambda, digits=2)
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]	[,13]
[1,]	**4.71**	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[2,]	0.00	**2.5**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[3,]	0.00	0.0	**1.45**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[4,]	0.00	0.0	0.00	**0.92**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[5,]	0.00	0.0	0.00	0.00	**0.85**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[6,]	0.00	0.0	0.00	0.00	0.00	**0.64**	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[7,]	0.00	0.0	0.00	0.00	0.00	0.00	**0.55**	0.00	0.00	0.00	0.00	0.00	0.0
[8,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	**0.35**	0.00	0.00	0.00	0.00	0.0
[9,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	**0.29**	0.00	0.00	0.00	0.0
[10,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.25**	0.00	0.00	0.0
[11,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.23**	0.00	0.0
[12,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.17**	0.0
[13,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.1**

Separation of Wines

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]	[,13]
[1,]	**4.71**	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[2,]	0.00	**2.5**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[3,]	0.00	0.0	**1.45**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[4,]	0.00	0.0	0.00	**0.92**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[5,]	0.00	0.0	0.00	0.00	**0.85**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[6,]	0.00	0.0	0.00	0.00	0.00	**0.64**	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[7,]	0.00	0.0	0.00	0.00	0.00	0.00	**0.55**	0.00	0.00	0.00	0.00	0.00	0.0
[8,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	**0.35**	0.00	0.00	0.00	0.00	0.0
[9,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	**0.29**	0.00	0.00	0.00	0.0
[10,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.25**	0.00	0.00	0.0
[11,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.23**	0.00	0.0
[12,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.17**	0.0
[13,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.1**

- It turns out that this matrix **provides us with the answer to our problem**:
- Any (normed) linear combination of the 13 standardized characteristics will also be a (normed) linear combination of the 13 uncorrelated eigenvalues;
- A little consideration shows that because of this, the variance of any normed linear combination can never exceed the maximum variance of the eigenvectors – **4.71**.
- The solution is thus **the first eigenvector**: `T[,1]`.

Separation of Wines

- The combination of the scaled data that varies the most:

```
round(T[,1],digits=2)
 [1] -0.14  0.25  0.00  0.24 -0.14 -0.39 -0.42  0.30 -0.31  0.09 -0.30
[12] -0.38 -0.29
```

- Thus the most varying combination of the scaled data is

$$-0.14 * \widetilde{Alcohol} + 0.25 * \widetilde{MalicAcid} + 0 * \widetilde{Ash} + 0.24 * \widetilde{AlcAsh} - 0.14 * \widetilde{Mg} - 0.39 * \widetilde{Phenols} - 0.42 * \widetilde{PhenolsFlav} + 0.30 * \widetilde{NonFlavPhenols} - 0.31 * \widetilde{Proa} + 0.09 * \widetilde{Color} - 0.30 * \widetilde{Hue} - 0.38 * \widetilde{OD} - 0.29 * \widetilde{Proline}$$

Where the $\sim$ versions are the scaled variables, with the mean subtracted and divided by the standard deviation.

- This is the scale that we want to look at, when we want to maximize the separation of wines.

Separation of Wines

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]	[,13]
[1,]	**4.71**	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[2,]	0.00	**2.5**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[3,]	0.00	0.0	**1.45**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[4,]	0.00	0.0	0.00	**0.92**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[5,]	0.00	0.0	0.00	0.00	**0.85**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[6,]	0.00	0.0	0.00	0.00	0.00	**0.64**	0.00	0.00	0.00	0.00	0.00	0.00	0.0
[7,]	0.00	0.0	0.00	0.00	0.00	0.00	**0.55**	0.00	0.00	0.00	0.00	0.00	0.0
[8,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	**0.35**	0.00	0.00	0.00	0.00	0.0
[9,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	**0.29**	0.00	0.00	0.00	0.0
[10,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.25**	0.00	0.00	0.0
[11,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.23**	0.00	0.0
[12,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.17**	0.0
[13,]	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.1**

- **Further conclusions:**

- The scale, uncorrelated with the first eigenvector, that varies the most, is exactly the 2nd eigenvector `T[,2]`, with variance **2.5**. And so on…

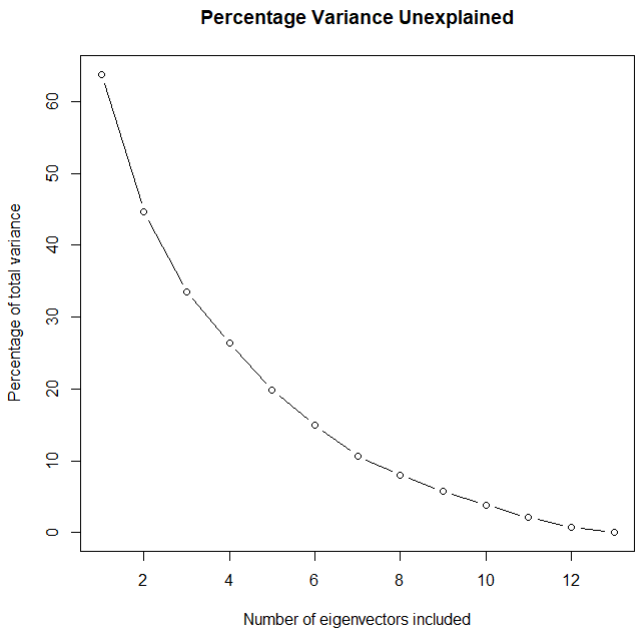
- The total variation after the coordinate shift is unchanged:

```
sum(diag(Lambda))
[1] 13
```

- Also note that the contribution from the 13th eigenvector is only 0.1/13, **0.7%**

Separation of Wines

```
plot(100*(13-cumsum(diag(Lambda)))/13,type="b",
     main="Percentage Variance Unexplained",
     xlab='Number of eigenvectors included',
     ylab='Percentage of total variance')
```



Eigenvectors	% variance explained
1	36
2	55
3	67
4	74
5	80
6	85
7	89
8	92
9	94
10	96
11	98
12	99
13	100

Principal Component Analysis

- PCA is a method to handle many variables, which are mutually correlated.
- PCA seeks to identify underlying dimensions in a data material, and to estimate the relationship between these.
- PCA may be used as a data reducing method, often in relation to multiple regression.
- PCA reduces the number of explanatory variables to a lesser number of ”principal komponents”, with (we hope) nearly as much of the variation as the inital variables.

What PCA does is essentially the contents of the preceding slides!

- The eigenvectors on the previous slides are **exactly** the Principal Components.

Identification of Lower- Dimensional Spaces

Figure 3.10 Data swarm (quasi-planar)

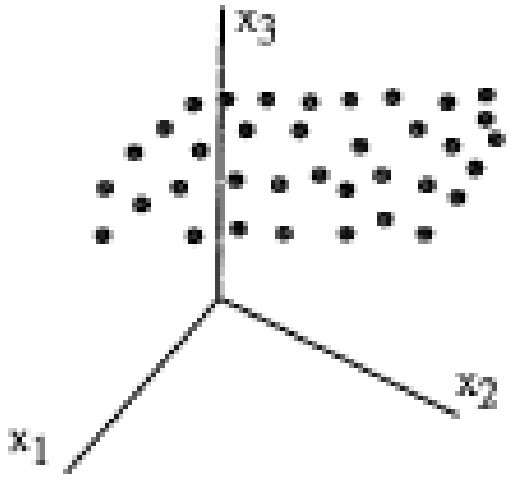
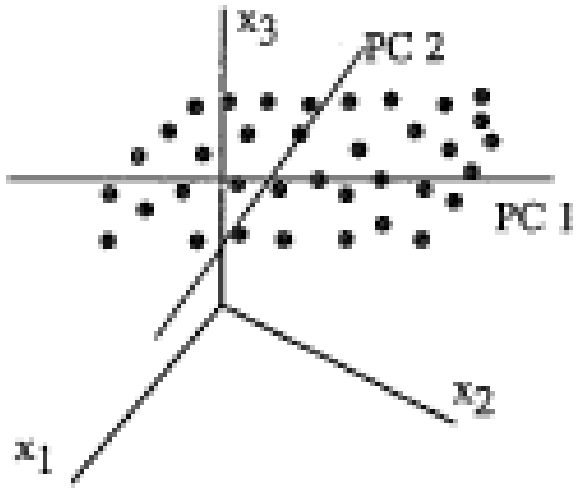


Figure 3.11 Data swarm with 2 PCs



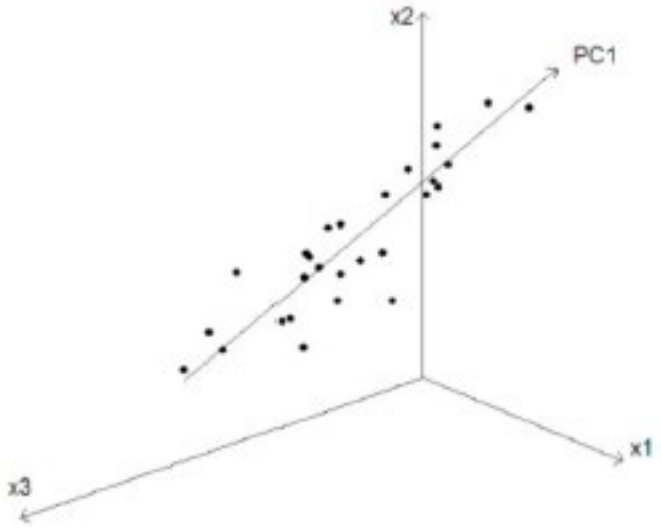
The First Principal Component – PC1

We look for the direction that **explains as much as possible of the variation in the data**. We assume here 3 variables and n=28:

$$\boldsymbol{p_1} = t_{11}\boldsymbol{X_1} + t_{21}\boldsymbol{X_2} + t_{31}\boldsymbol{X_3},$$

where $\sum_{j=1}^{3} {t_{1j}}^2 = 1$.

- p are the ”scores”;
- t are the ”loadings”.



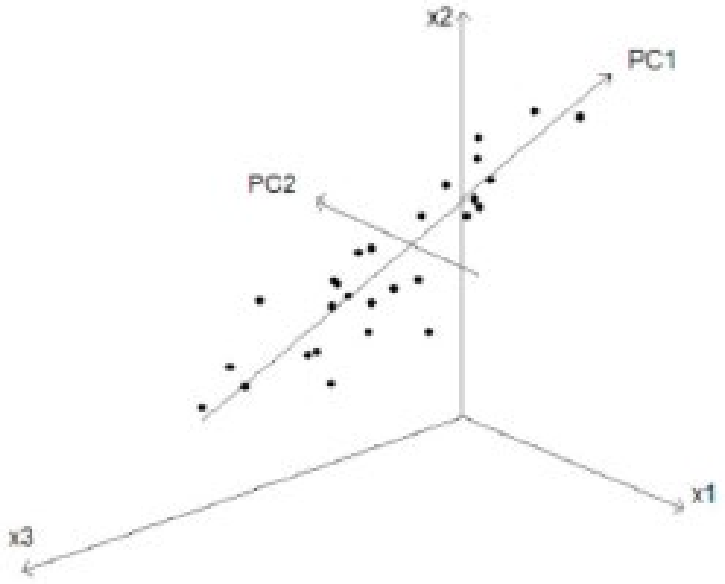
The Second Principal Component – PC2

We consider the plane perpendicular to **PC1**, and find the linear combination the explans the 2nd most variation:

$$\boldsymbol{p_2} = t_{12}\boldsymbol{X_1} + t_{22}\boldsymbol{X_2} + t_{32}\boldsymbol{X_3},$$

where $\sum_{j=1}^{3} {t_{2j}}^2 = 1$

PC1 og PC2 and are orthogonal.



PCA of the Wines Data

```
wines.PC<- PCA(scale(wines))
names(wines.PC)
[1] "scores"        "loadings"      "var"           "totalvar"
[5] "centered.data"
```

```
summary(wines.PC)

PCA model of a mean-centered matrix of 178 by 13
Number of PCs to cover 90 percent of the variance: 8

             Var Cumul. var.
PC 1   36.198848    36.19885
PC 2   19.207490    55.40634
PC 3   11.123631    66.52997
PC 4    7.069030    73.59900
PC 5    6.563294    80.16229
PC 10   1.930019    96.16972
```

- Lets take a look at the components

PCA of the Wines Data – the Loadings

- The **loadings** are the *coordinates of the principal components*:

```
> head(wines.PC$loadings,n=3)
                 PC 1       PC 2        PC 3        PC 4        PC 5       PC 6
Alcohol   -0.144329395 0.4836515 -0.20738262  0.0178563 -0.26566365 0.2135386
MalicAcid  0.245187580 0.2249309  0.08901289 -0.5368903  0.03521363 0.5368138
Ash        0.002051061 0.3160688  0.62622390  0.2141756 -0.14302547 0.1544747
                PC 7        PC 8        PC 9       PC 10       PC 11       PC 12
Alcohol   -0.05639636  0.39613926 -0.50861912  0.21160473  0.22591696 -0.26628645
MalicAcid  0.42052391  0.06582674  0.07528304 -0.30907994 -0.07648554  0.12169604
Ash       -0.14917061 -0.17026002  0.30769445 -0.02712539  0.49869142 -0.04962237
                PC 13
Alcohol    0.01496997
MalicAcid  0.02596375
Ash       -0.14121803
> head(T,n=3)
             [,1]       [,2]        [,3]       [,4]        [,5]       [,6]        [,7]
[1,] -0.144329395 -0.4836515 -0.20738262 -0.0178563  0.26566365 0.2135386  0.05639636
[2,]  0.245187580 -0.2249309  0.08901289  0.5368903 -0.03521363 0.5368138 -0.42052391
[3,]  0.002051061 -0.3160688  0.62622390 -0.2141756  0.14302547 0.1544747 -0.14917061
            [,8]        [,9]       [,10]       [,11]       [,12]       [,13]
[1,]  0.39613926  0.50861912  0.21160473 -0.22591696 -0.26628645 -0.01496997
[2,]  0.06582674 -0.07528304 -0.30907994  0.07648554  0.12169604 -0.02596375
[3,] -0.17026002 -0.30769445 -0.02712539 -0.49869142 -0.04962237  0.14121803
```

- The principal components are only identified up to a sign change.

PCA of the Wines Data – the Scores

- The **scores** are the *new coordinates* of the (scaled) wines data *relative to the principal components*:

```
head(wines.PC$scores,n=3)
        PC 1       PC 2       PC 3       PC 4       PC 5       PC 6       PC 7
[1,] -3.307421  1.4394023 -0.1652728  0.2150246  0.6910933  0.2232504 0.59474883
[2,] -2.203250 -0.3324551 -2.0207571  0.2905387 -0.2569299  0.9245123 0.05362434
[3,] -2.509661  1.0282507  0.9800541 -0.7228632 -0.2503270 -0.5477310 0.42301218
          PC 8       PC 9     PC 10     PC 11       PC 12       PC 13
[1,] -0.06495586 -0.6396384 1.0180840 0.4502932 0.5392891439 -0.066052305
[2,] -1.02153432  0.3079780 0.1592521 0.1422560 0.3871456499  0.003626273
[3,]  0.34324787  1.1745213 0.1130420 0.2858665 0.0005819316  0.021655423
> head(scale(wines)%*%T,n=3)
        [,1]       [,2]       [,3]       [,4]       [,5]       [,6]       [,7]
[1,] -3.307421 -1.4394023 -0.1652728 -0.2150246 -0.6910933  0.2232504 -0.59474883
[2,] -2.203250  0.3324551 -2.0207571 -0.2905387  0.2569299  0.9245123 -0.05362434
[3,] -2.509661 -1.0282507  0.9800541  0.7228632  0.2503270 -0.5477310 -0.42301218
          [,8]       [,9]     [,10]      [,11]       [,12]       [,13]
[1,] -0.06495586  0.6396384 1.0180840 -0.4502932 0.5392891439  0.066052305
[2,] -1.02153432 -0.3079780 0.1592521 -0.1422560 0.3871456499 -0.003626273
[3,]  0.34324787 -1.1745213 0.1130420 -0.2858665 0.0005819316 -0.021655423
```

- Note the same sign changes

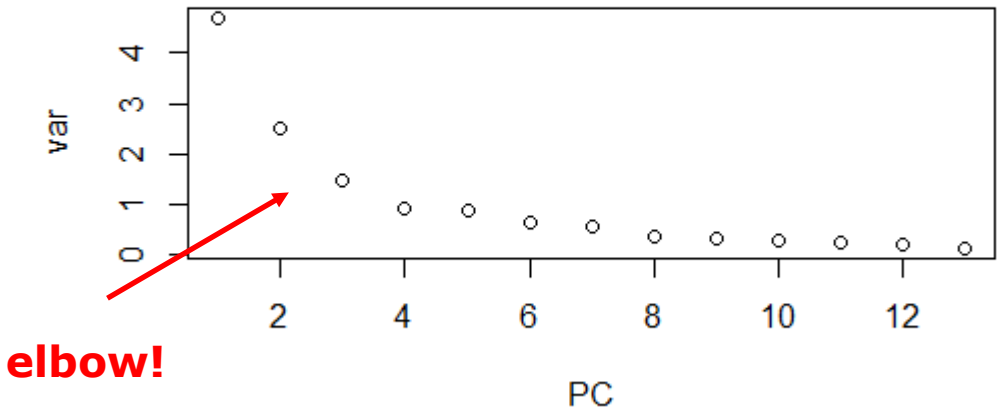
PCA of the Wines data – var, totalvar and centered.data

```
wines.PC$var
     PC 1      PC 2      PC 3      PC 4      PC 5      PC 6      PC 7      PC 8
4.7058503 2.4969737 1.4460720 0.9189739 0.8532282 0.6416570 0.5510283 0.3484974
     PC 9     PC 10     PC 11     PC 12     PC 13
0.2888799 0.2509025 0.2257886 0.1687702 0.1033779

wines.PC$totalvar
[1] 13
wines.PC$centered.data
[1] TRUE
```

We recognize the eigenvalues of the matrix Λ, and the sum of these. Lastly an indicator that we have 'done the right thing' (in this case).

Selecting the Number of Principal Components – the Skree Plot

```
plot(1:13,wines.PC$var,xlab="PC",ylab="var")
```



- Rules of thumb:

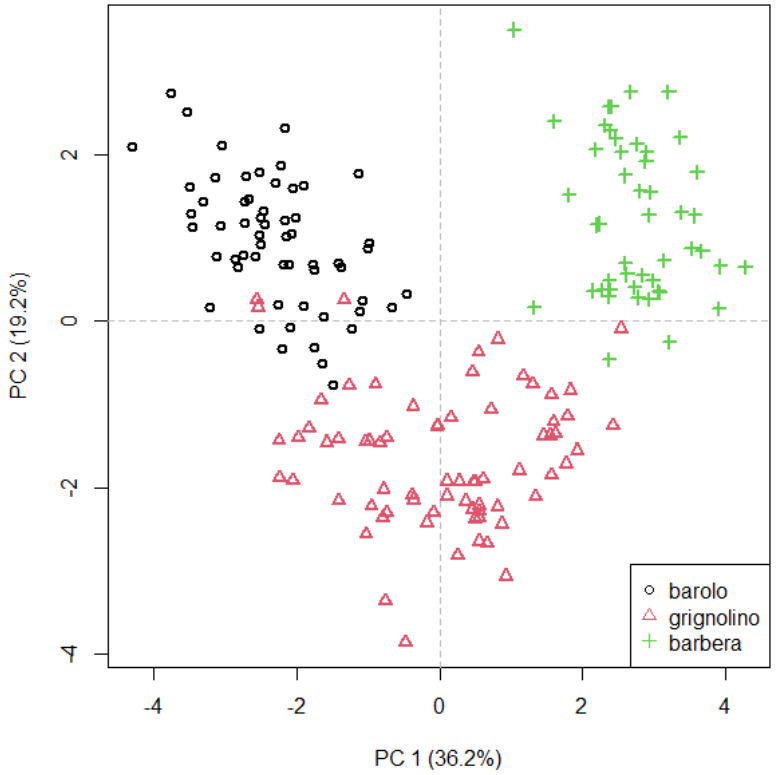
i. You select a number of principal components where the 'elbow' of the graph is.

ii. You usually (but not always) only select principal components with a variance greater than 1 – if the value is lower, the PC explain less than one average ordinary observation.

iii. You fix the amount of total variation that you need explained – t.ex. 80%.

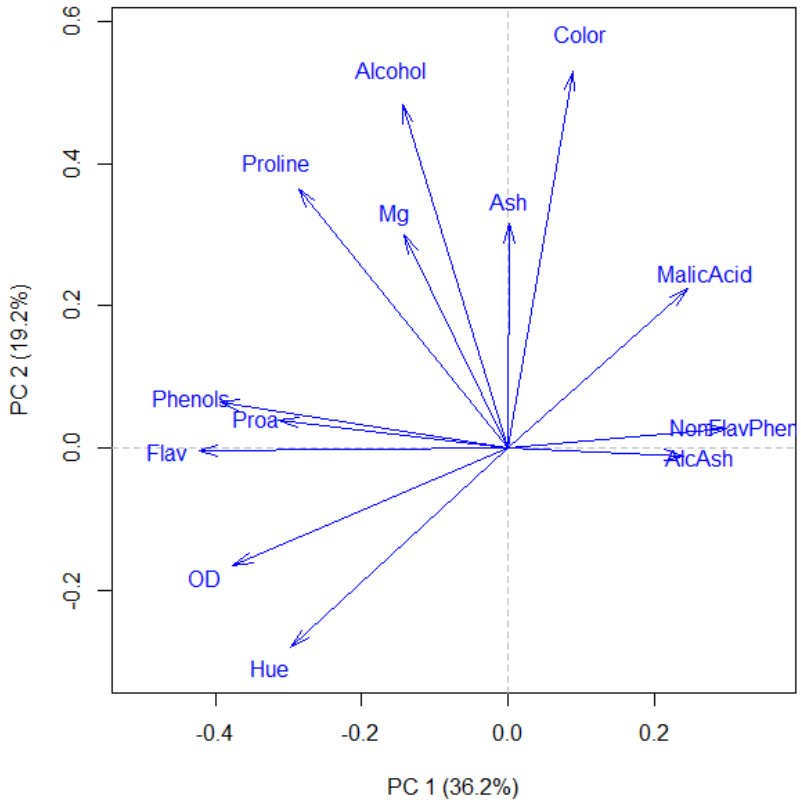
The Score plot

```
scoreplot(wines.PC, col = vintages, pch= as.numeric(vintages), lwd=2)
legend("bottomright",levels(vintages), col=1:3,pch=1:3)
```



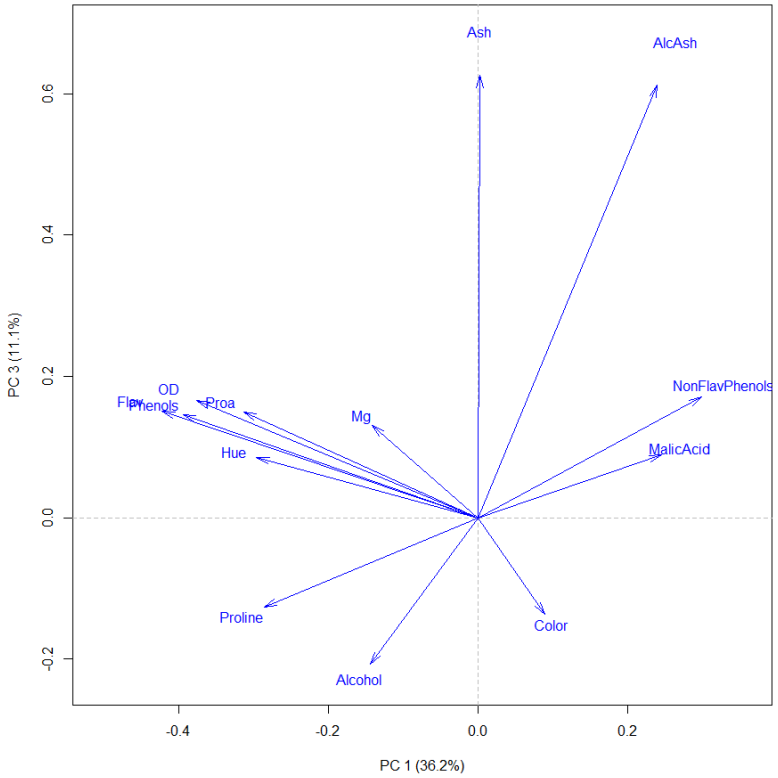
The Loading plot

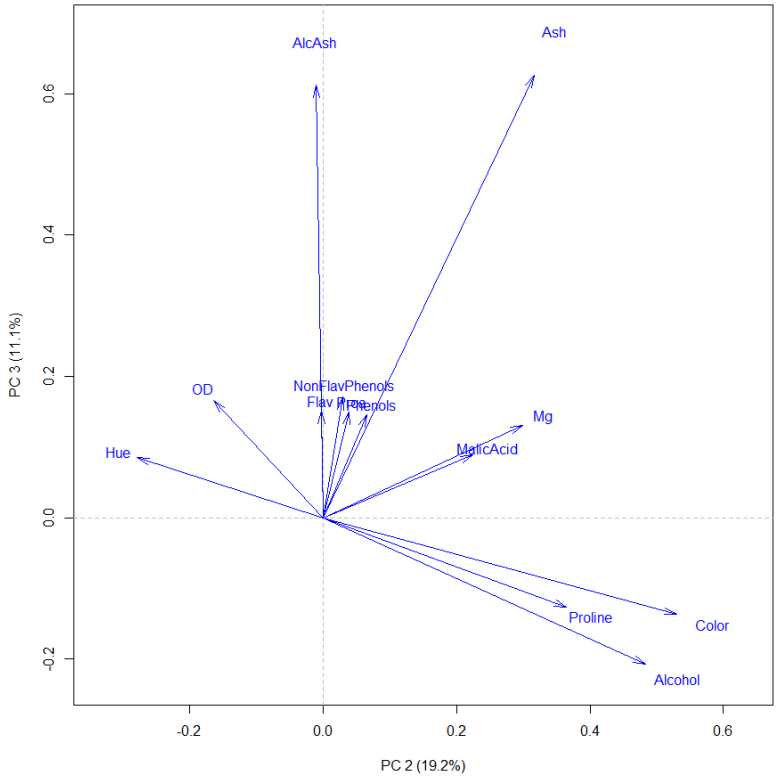
```
loadingplot(wines.PC, show.names= TRUE)
```



Higher Order Loading Plots

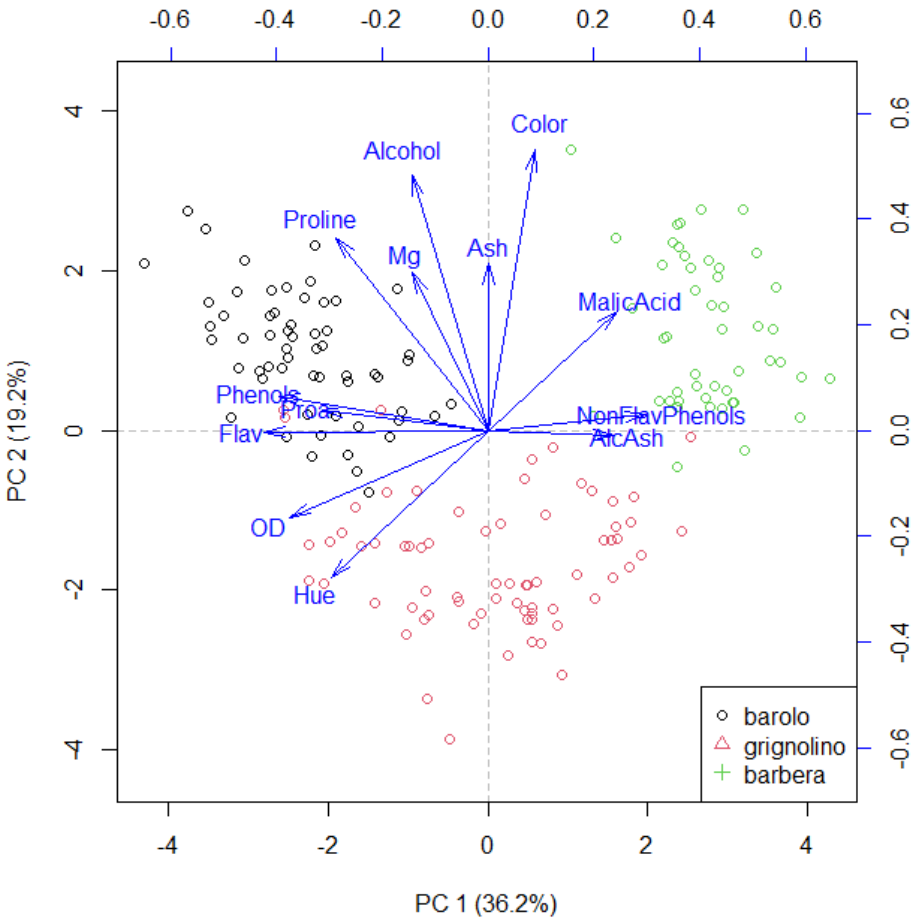
```
par(mfrow=c(1,2))
loadingplot(wines.PC, pc=c(1,3), show.names= TRUE)
loadingplot(wines.PC, pc=c(2,3), show.names= TRUE)
```





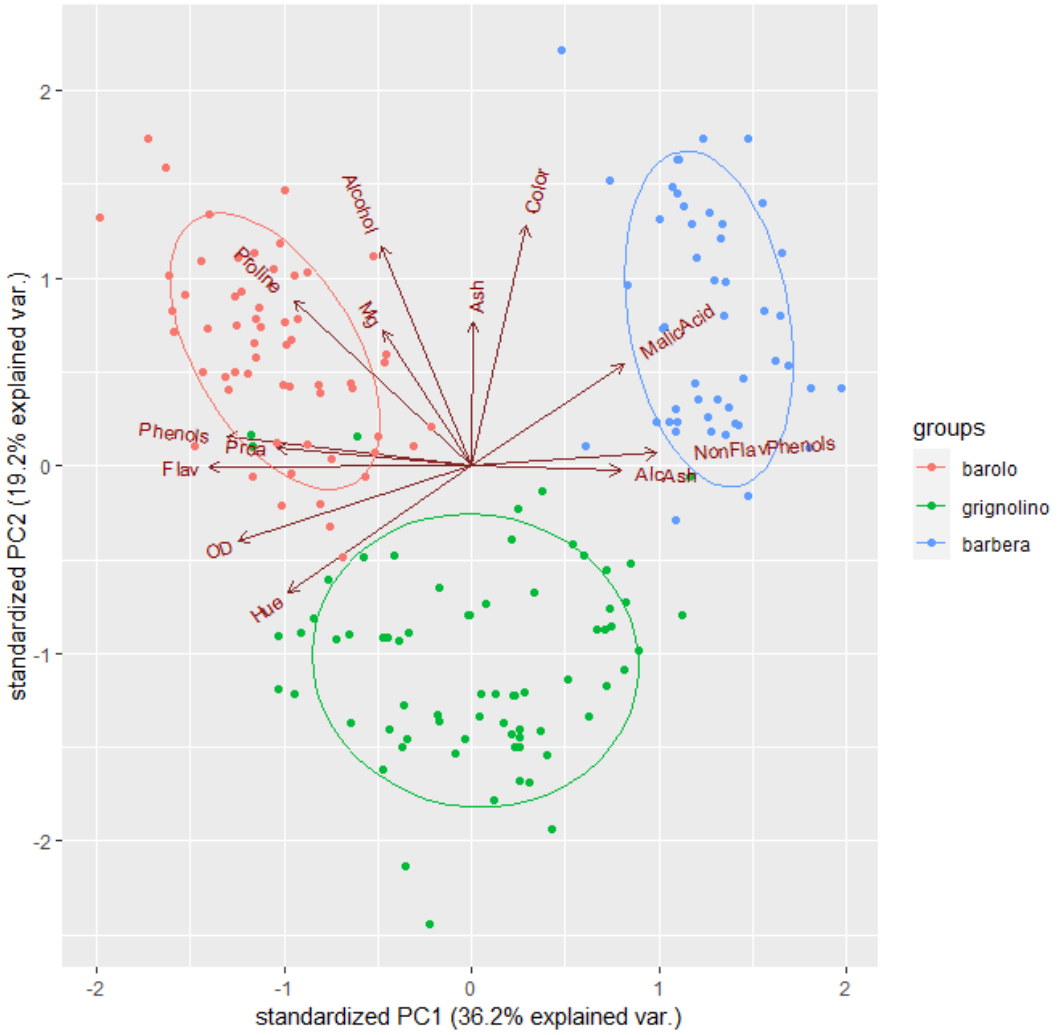
The Biplot

```
biplot(wines.PC, score.col = vintages, show.names = "loadings")
legend("bottomright",levels(vintages), col=1:3,pch=1:3)
```



Alternative Biplot

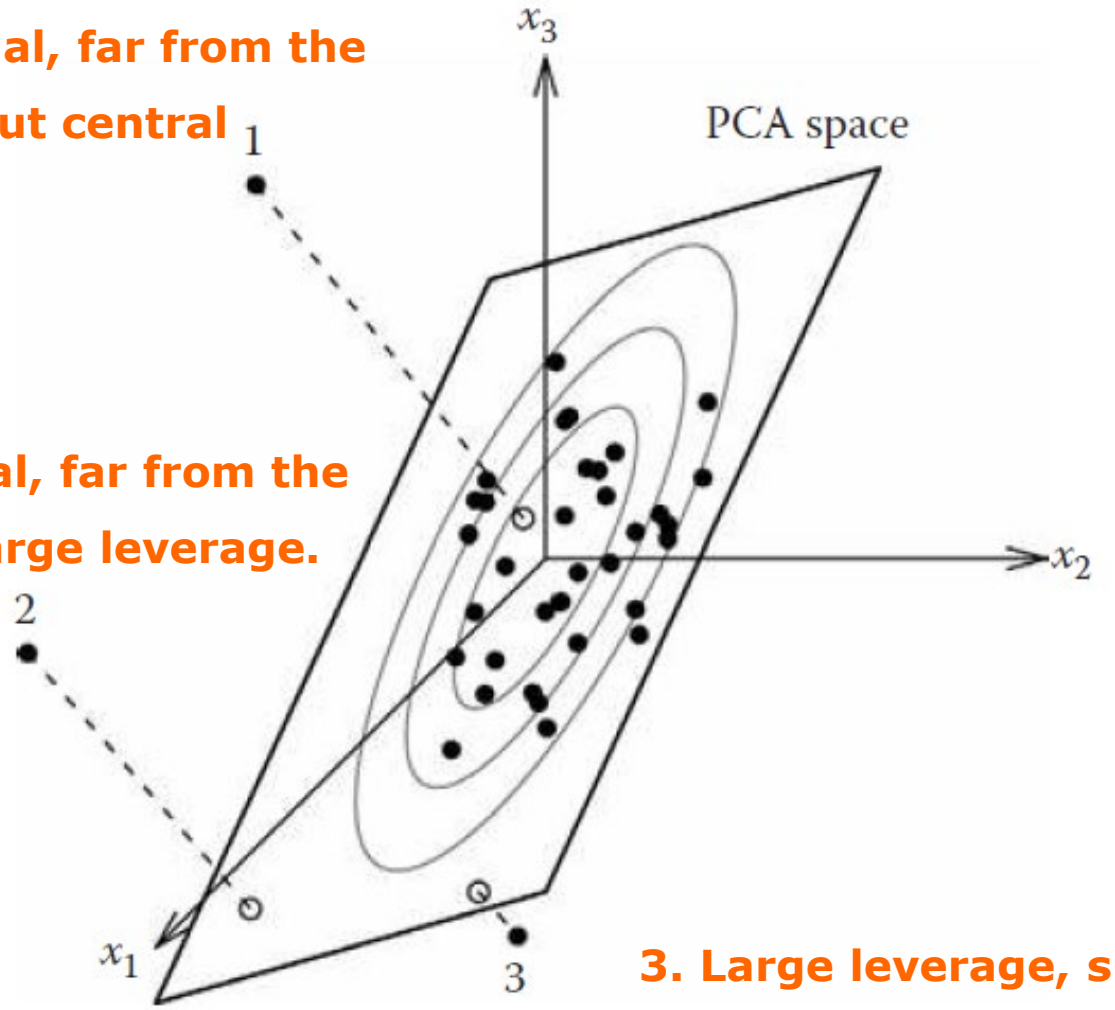
```
ggbiplot(prcomp(scale(wines)),groups=vintages,ellipse=T)
```



Diagnostic Plots – Residuals and Leverage

1. Large residual, far from the PCA space but central

2. Large residual, far from the PCA space, large leverage.

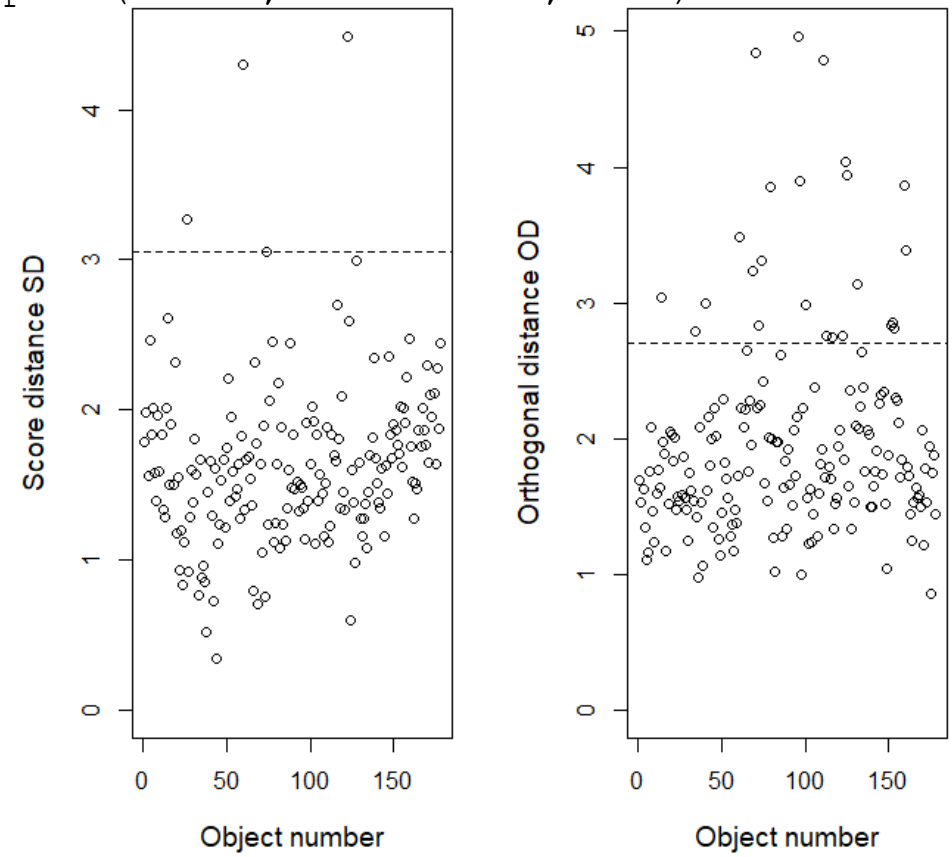


3. Large leverage, small residual

Diagnostic Plots

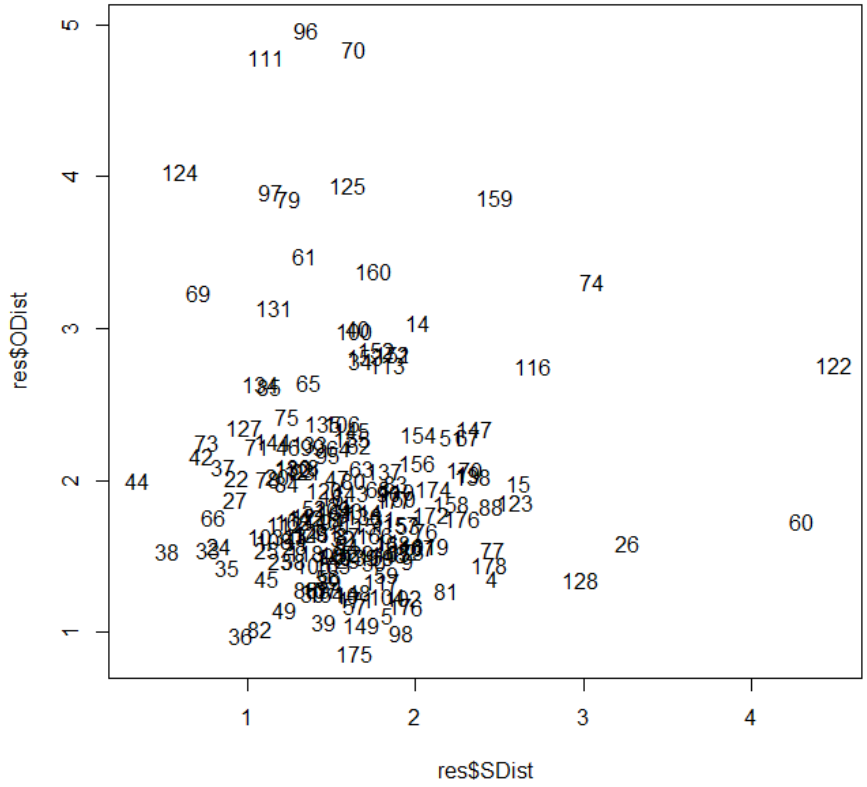
Plotting distances for 3 principal components, leverage (left) and residual (right):

```
wines.PCA<- princomp(wines, cor = TRUE)
res<-pcaDiagplot(wines, wines.PCA, a=3)
```



Diagnostic Plots

```
par(mfrow=c(1,1))
plot(res$SDist, res$ODist, type="n")
text(res$SDist, res$ODist, labels=as.character(1:178))
```



No points have high leverage and also high residual

Application Areas for PCR

Areas with complex, correlated data structures:

- Quantitative Finance;
- Neuroscience
- Spectroscopy(UV-VIS, x-ray, IR, NIR, NMR etc.)
- Questionaires
- Image data
- Etc.

Example: Jam

- Results from a taste survey for jam. 12 observations of raspberry jam, where berries are picked at four locations C1, C2, C3, C4, and harvested at three time points H1, H2, H3.

- A response variable Y is an averaged preference score from 0 to 9, given by 114 repræsentative consumers.

- Explanatory variables are twelve sensoric variables, where trained sensoric panel members have evaluated these:

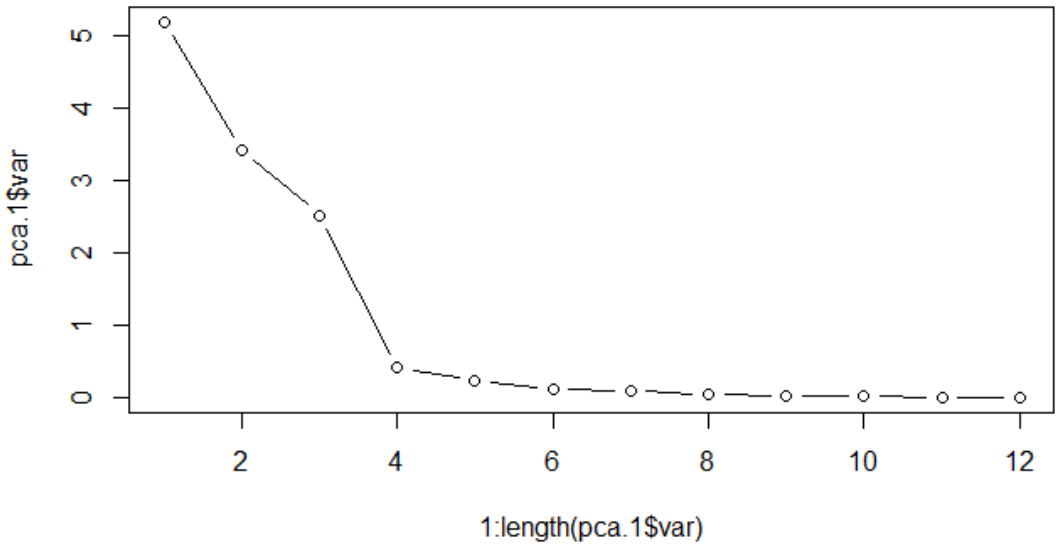
1) REDNESS	Redness	7) SOURNESS	Sourness
2) COLOUR	Colour intensity	8) BITTERNE	Bitterness
3) SHININES	Shininess	9) OFF.FLAV	Off-flavour
4) R.SMELL	Raspberry smell	10) JUICINES	Juiciness
5) R.FLAV	Raspberry taste	11) THICKNES	Thickness
6) SWEETNES	Sweetness	12) CHEW:RES	Chewing resistance

Performing PCA

```
jam<-read.table("Data/Jam.txt", header=TRUE, quote="\"")
# first column is names, last column is outcome:
pca.1<- PCA(scale(jam[ ,-c(1,14)]))
summary(pca.1)

PCA model of a mean-centered matrix of 12 by 12
Number of PCs to cover 90 percent of the variance: 3

              Var Cumul. var.
PC 1  43.11404052    43.11404
PC 2  28.45577680    71.56982
PC 3  20.92326359    92.49308
PC 4   3.41038319    95.90346
PC 5   1.92478143    97.82825
PC 10  0.05286637    99.99396
>
```

Example: Jam - Skree Plot

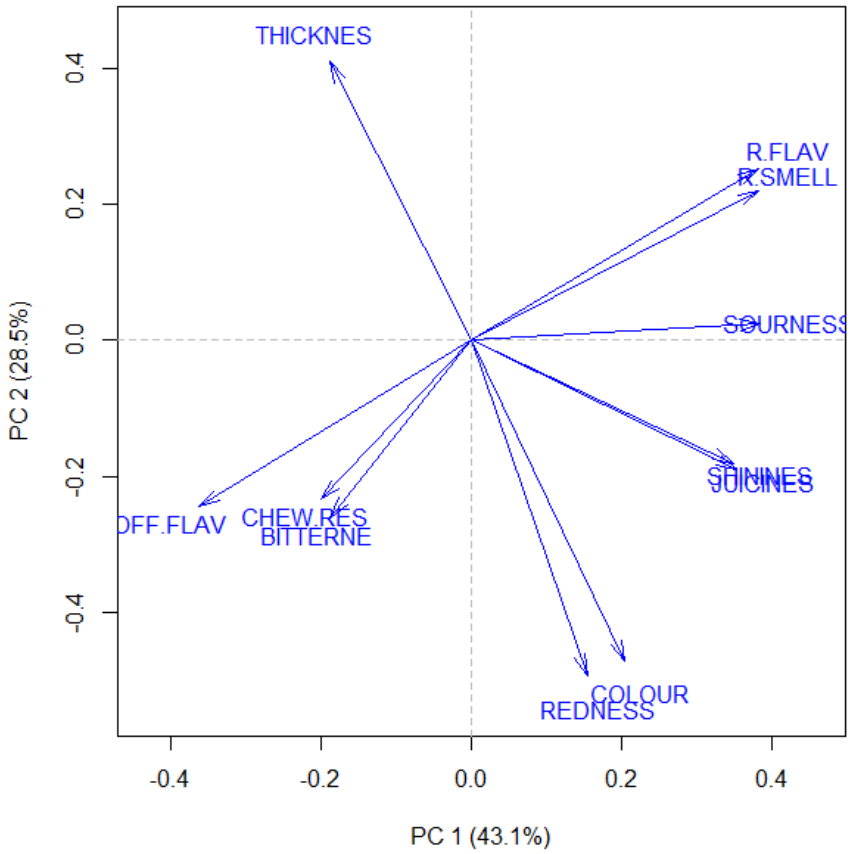
```
plot(1:length(pca.1$var), pca.1$var, type="b")
```



- We choose 3 principal components

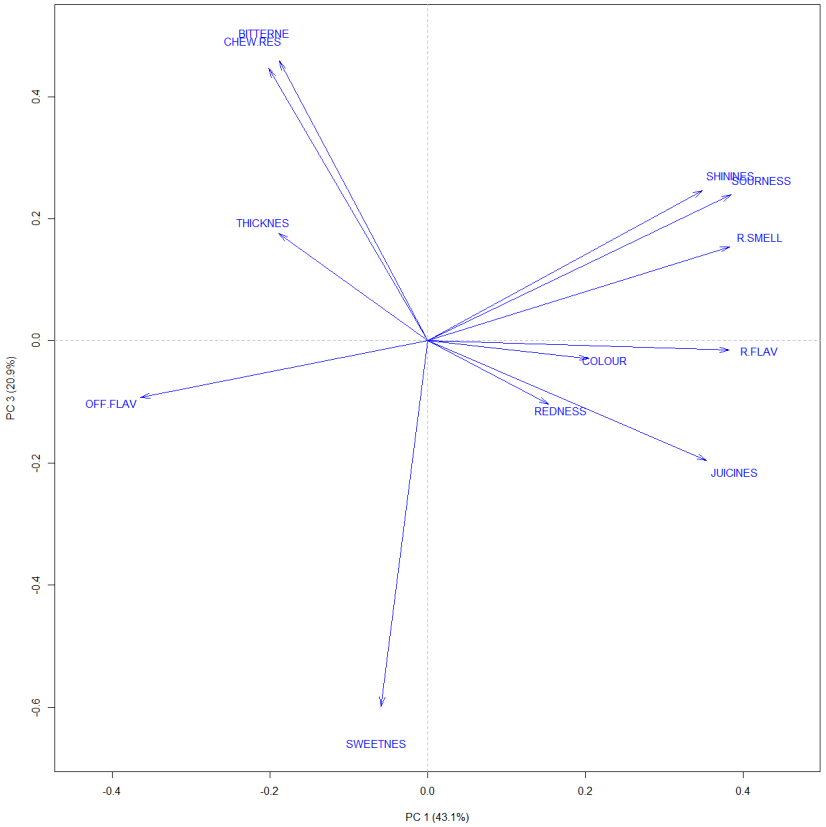
Example: Jam – Loading plot

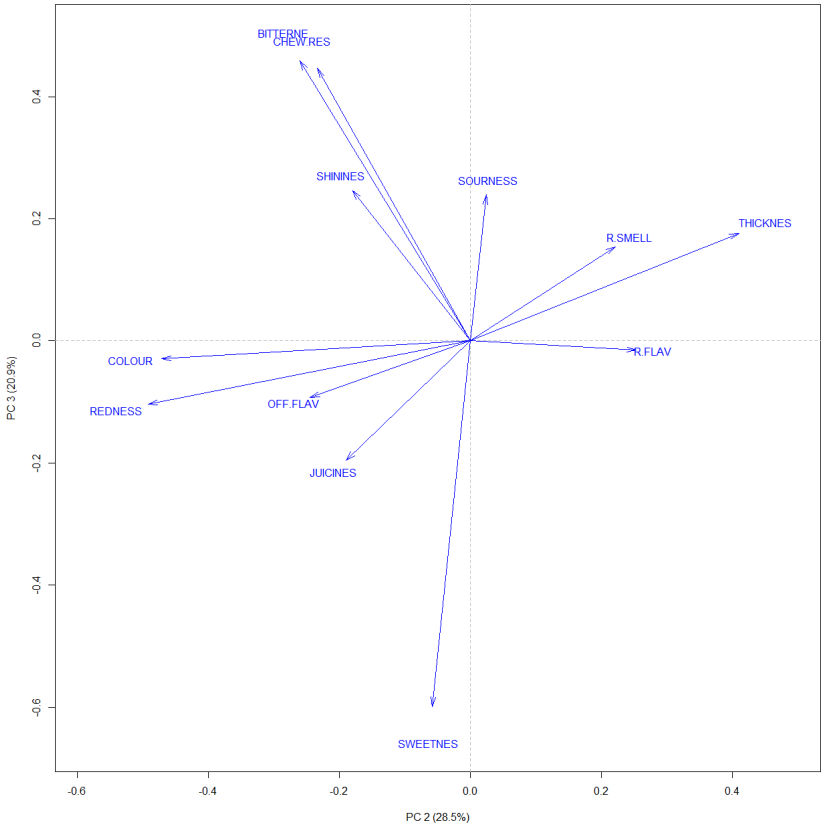
```
loadingplot(pca.1, show.names= TRUE)
```



Example: Jam – Loading plot

```
par(mfrow=c(1,2))
loadingplot(pca.1, pc=c(1,3), show.names= TRUE); loadingplot(pca.1, pc=c(2,3), show.names= TRUE)
par(mfrow=c(1,1))
```





Example: Jam – Loadings. Labelling the PCs

```
>loadings(pca.1)

Loadings:
            PC 1   PC 2   PC 3
REDNESS    0.154 -0.492 -0.104
COLOUR     0.204 -0.472 -0.029
SHININES   0.349 -0.180  0.246
R.SMELL    0.383  0.221  0.154
R.FLAV     0.382  0.253 -0.015
SWEETNES -0.059 -0.058 -0.599
SOURNESS   0.385  0.024  0.239
BITTERNE -0.188 -0.260  0.459
OFF.FLAV -0.364 -0.245 -0.093
JUICINES   0.353 -0.190 -0.195
THICKNES -0.189  0.409  0.176
CHEW.RES -0.201 -0.234  0.446
```

PC1: Raspberry Feeling

PC2: Looks and consistency

PC3: Taste experience;

sweetness (-) / bitterness (+)/ chewing resistance (+)

PCR – Principal Component Regression
Linear Regression of Response on PCs

- We wish to regress the Response variable Y, the averaged preference score, on scores of Raspberry feeling, Looks and Consistency, and Taste Experience

- We will skip the selection procedure for number of PCAs. In PCR, the selection of the number of PCs needs to involve the response through cross-validation. We will stick to 3 PCs.

PCR – Principal Component Regression
Linear Regression of Response on PCs

- We use the three PCS as explanatory variables in a multiple regression model, with Y as response.

- We thus consider the following model:

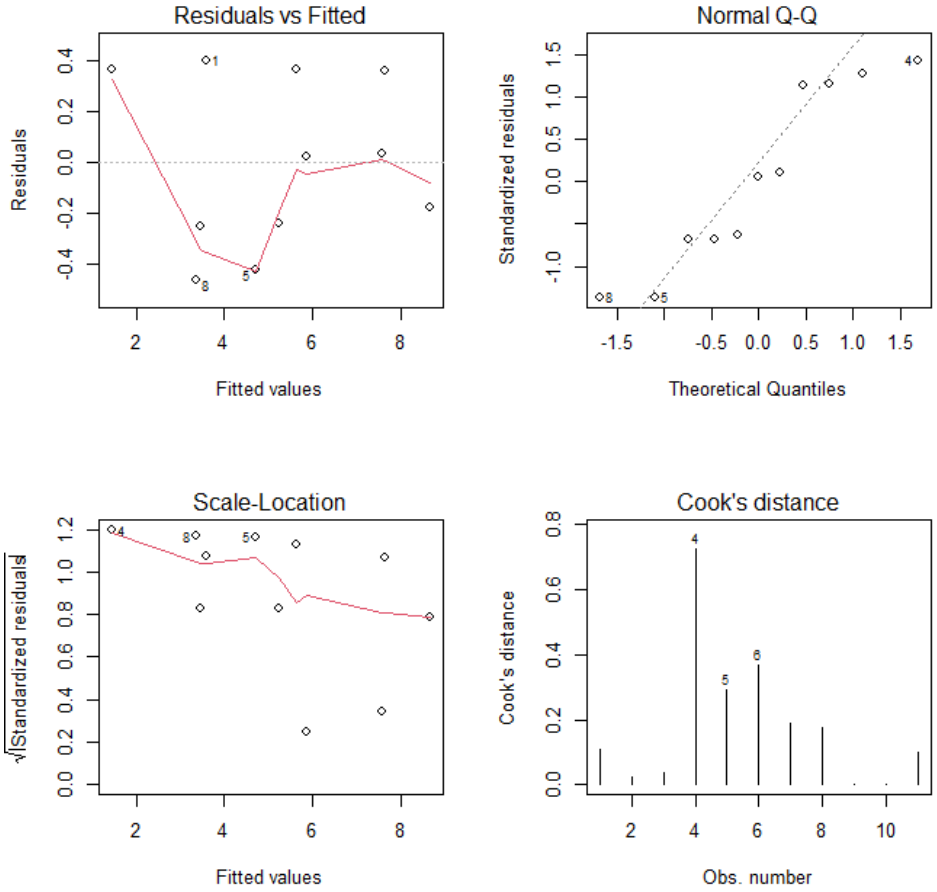
$$Y_i = \beta_0 + \beta_1 \cdot PC1_i + \beta_2 \cdot PC2_i + \beta_3 \cdot PC3_i + \varepsilon_i$$

where Y_i is the preference, and $PC1_i$, $PC2_i$ and $PC3_i$ are the principal component scores: our new, known and labeled explanatory variables. Finally, we have $\varepsilon_i \sim N(0, \sigma^2)$.

PCR – Principal Component Regression
Linear Regression of Response on PCs

Leaving out an outlier:

```
scores<-pca.1$scores
analysis<-lm(jam$PREFEREN[-12] ~
 . , as.data.frame(scores[-12 ,1:3]))

par(mfrow=c(2,2))
plot(analysis,which=1:4)
par(mfrow=c(1,1))
```



PCR – Principal Component Regression
Linear Regression of Response on PCs

Model Reduction:

```
drop1(analysis,test="F")
Single term deletions

Model:
jam$PREFEREN[-12] ~ `PC 1` + `PC 2` + `PC 3`
       Df Sum of Sq    RSS     AIC  F value    Pr(>F)
<none>               1.093 -17.396
`PC 1`  1    0.5154  1.609 -15.148   3.2998    0.1121
`PC 2`  1   31.1115 32.205  17.816 199.2083 2.126e-06 ***
`PC 3`  1   15.7846 16.878  10.709 101.0694 2.066e-05 ***

analysis<-lm(jam$PREFEREN ~  . , as.data.frame(scores[,2:3]))
drop1(analysis,test="F")
Single term deletions

Model:
jam$PREFEREN ~ `PC 2` + `PC 3`
       Df Sum of Sq    RSS     AIC F value    Pr(>F)
<none>               5.148 -4.1546
`PC 2`  1    30.169 35.317 16.9536  52.739 4.754e-05 ***
`PC 3`  1    17.879 23.028 11.8215  31.255 0.0003383 ***
```

PCR – Principal Component Regression
Linear Regression of Response on PCs

```
> summary(analysis)$coef
             Estimate Std. Error   t value     Pr(>|t|)
(Intercept)  5.0166667  0.2183356 22.976864 2.662062e-09
`PC 2`      -0.8962075  0.1234079 -7.262157 4.753719e-05
`PC 3`      -0.8045862  0.1439174 -5.590609 3.383175e-04
```

- Looks and Consistency and Taste Experience has an impact on the preference, while Raspberry Feeling does not;
- PC2: Evaluators appreciate **Redness** and **Colour**, but not **Thickness** of berries.
- PC3: Evaluators appreciate **Sweetness**, but not **Bitterness** and **Chewing Resistance**.

Example: Image Analysis

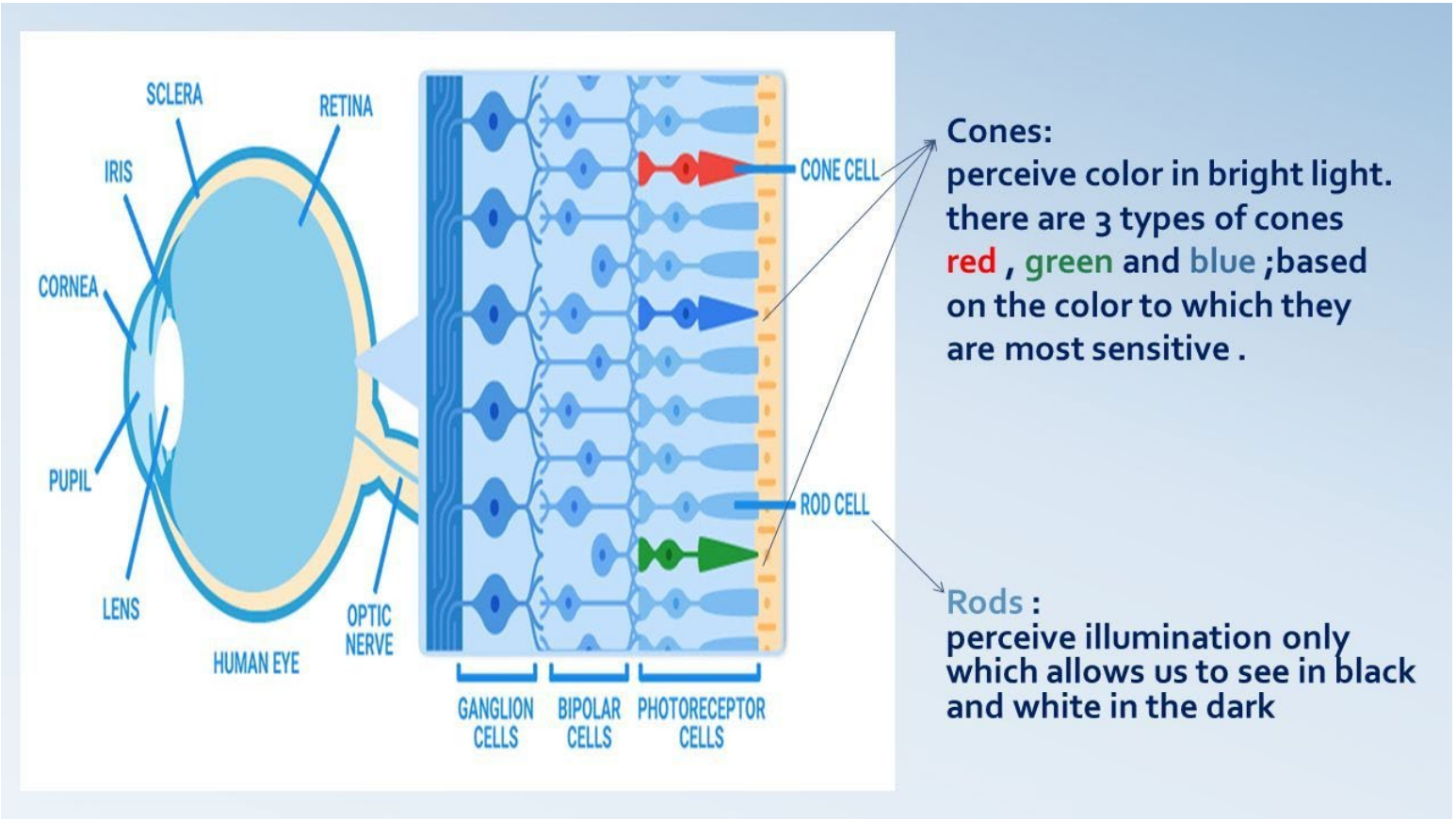
Horse Analysis

```
horse <- readJPEG("Data/horse.jpg")

ncol(horse)
[1] 480

nrow(horse)
[1] 341

#480*341 pixels - 480*341*3=491.040 numbers


# array with 3 layers:
str(horse)
 num [1:341, 1:480, 1:3] 0.796 0.796 0.796 0.796 0.796 ...
```

- Why 3 layers? RGB-coding; **Red-Green-Blue**.

PCA Example: Image compression

Color pictures are coded with an intensity of **Red**, **Green** and **Blue** :



Horse Analysis

```
str(horse)
num [1:341, 1:480, 1:3] 0.796 0.796 0.796 0.796 0.796 ...

red  <- horse[,,1]
green<- horse[,,2]
blue <- horse[,,3]
```

Running PCA on red, green and blue
(with `prcomp() for compatibility`):

```
horse.red.pca <- prcomp(red, center = FALSE)
horse.green.pca<- prcomp(green, center = FALSE)
horse.blue.pca <- prcomp(blue, center = FALSE)
```

Gather PCA objects in one list:

```
rgb.pca <- list(horse.red.pca, horse.green.pca, horse.blue.pca)
```

Horse Analysis

Indexing after the amount of principal components (max is the number of rows in the picture – 341):

```
>index<-c(3,6,9,12,15,18,50,100)
```

The function below reconstructs the picture from the first i principal components;
'x' are the principal components, 'rotation' are the loadings (se `?prcomp`), such that what is returned is the inversion from scores to standard coordinates (scores%*%T^{-1}) which constitute the reconstruction. If T is all the PCs, the originale picture is obtained.

```
my.reconstruct<-function(j) {
  return( j$x[,1:i] %*% t(j$rotation[,1:i]))
  }
```

Create picture based on the first i principal components:

```
for (i in index) {
  pca.picture <- sapply(rgb.pca,my.reconstruct,simplify = 'array')
  writeJPEG(pca.picture,
    paste("picture/horse_compressed_",i,"_components.jpg", sep =""))
  }
```

Horse Analysis – Results

3 PCs

6 PCs

9 PCs

12 PCs

15PCs

18PCs

50PCs

100PCs

Original

Horse Analysis – Conclusion

- Further immediate use of PCs above 100 does not change the picture quality. It is hard to tell the difference to the original with 341 components…
- The compression rate for the picture with 100 PCs is 58% (ratio of bit sizes).
- One can use the picture with 100PCs, and save 58% storage.