Multivariate statistics

27 marzo 2025

Indice

1 Introduction **5**
1.1 Matrices . . . 6
1.2 Giorno 2 . . . 10
1.3 Lab . . . 12

2 Cluster analysis **15**
2.1 Distance matrix . . . 15
2.1.1 Distances . . . 15
2.1.1.1 All numeric variables . . . 15
2.1.1.2 All binary variables . . . 16
2.1.1.3 All categorical variables . . . 18
2.1.1.4 Mixed types . . . 18
2.2 Clustering . . . 19
2.3 Hierarchical agglomerative methods . . . 20
2.3.1 Single linkage . . . 21
2.3.2 Complete linkage . . . 22
2.3.3 Average linkage . . . 22
2.3.4 Centroid methods . . . 23
2.3.5 Ward's method . . . 24
2.4 Partitioning methods (k-means) . . . 24
2.4.1 k-means clustering . . . 24
2.4.2 Other methods like k-means . . . 25
2.4.2.1 PAM . . . 25
2.4.2.2 Model based clustering . . . 26
2.4.3 Choice of k in k-means . . . 27
2.4.3.1 Silhouette score . . . 27
2.5 Lab . . . 29
2.6 K-means . . . 37

3 Principal component analysis **39**
3.1 Preliminaries . . . 39
3.2 PCA . . . 42
3.3 Properties . . . 46
3.4 Obtaining PCs in practice . . . 47
3.5 Interpretation . . . 51
3.6 Number of components to choose . . . 54
3.6.1 Percentage of explained variance . . . 55
3.6.2 Threshold on eigenvalues (Kaiser's rule) . . . 56

3.6.3 Screeplot 57
3.6.4 Correlation between PC and variables 59
3.7 Principal component scores 60
3.8 Importance of last components 60
3.8.1 Goodness of first PCs 61
3.8.2 Multicollinearity 61
3.9 Orthogonal least square 61
3.10 PCA and SVD 62
3.11 Lab 63

4 Factor analysis 83
4.1 Introduction 83
4.2 Assumptions 85
4.3 Factor loading matrix 88
4.4 Equivariance 89
4.5 Model identifiability 90
4.6 Estimation of the linear factor model 92
4.7 Principal factor method 94
4.8 ML method 98
4.9 Factor rotation 100
4.9.1 Variamax 101
4.10 Factor score estimation 102
4.10.1 Thompson 102
4.10.2 Bartlett estimator 103
4.11 Relationship between PCA and Factor Analysis 104
4.12 Lab 106
4.13 Example with dataset instead of correlation matrix 117
4.14 Last exercise 123

5 Discriminant analysis and supervised classification 127
5.1 Discriminant analysis 127
5.2 Doing classification 134
5.3 Classification based on probability models 136
5.4 Lab 140
5.4.1 Jobs 140
5.4.2 banknotes 148
5.4.3 wdbc 150

Capitolo 1

Introduction

Remark 1 (Prerequisites). random variables/vectors, linear models (least square principle, assumptions, R^2), matrix algebra (matrices, determinant, inverse, quandratic forms, eigenvalues-eigenvectors, lagrange multipliers)

Remark 2 (Multivariate analysis situation). Large number of variables (p) observed on the same set of statistical units (n). Aim is to summarize somehow the data.

Important remark 1. Methods tackled in the course:

- *dimension reduction methods*: among these we see principal components analysis and factor analysis. They share the aim to summarize the information contained in our data building new variables, limited in number, which can convey as much info as the original set. The two methods we see are:
  - *Principal component*: is data transformation method. We transform data to obtain meaningnful summary. Always feasible.
  - *Factor analysis*: is a model where we make assumptions, fit and check if the model adequately fix the data.
- *discriminant analysis*: a supervised classification method. We have two (or more than two) groups and know that groups are different by characteristics (eg healty/diseased) we collect the same varaibles for the two groups. We want to find
  - a way to characterize the two groups, how different are which respect to which characterisc
  - a criteria to separate the two groups; also find a rule to characterize and a ssign a new observation (whose group membership is unknown) to one or the other group.
- *cluster analysis*: previous variable have to be numeric, here of any kind and our aim is to find groups in the data (we have no groups as in discriminant analysis). Splitting should be homogeneous data within group and different between groups. (eg customer satisfaction: find groups that are equally satisfied, to send personalized advertisement)

We start from cluster analuysis and then the rest, but before that some matrices and notation that will be needed in future

1.1 Matrices

Data matrix Main ingredient of our course is data matrix, the one including all data on observation were interested in. We denote it as $\mathbf{X}$, which is $n \times p$ with n units (each row correspond to a different unit, the generic one is unit j) and p variables (each column correspond to a variable, the generic one being i).

$$\underset{n\times p}{\mathbf{X}} = \begin{bmatrix} x_{11} & \dots & x_{1i} & \dots x_{1p} \\ \dots & \dots & \dots & \dots \\ x_{j1} & \dots & x_{ji} & \dots x_{jp} \\ \dots & \dots & \dots & \dots \\ x_{n1} & \dots & x_{ni} & \dots x_{np} \end{bmatrix}$$

If we read rowwise we have the profile of each individual, while columnwise each column describe the univariate dataset (how each variable is expressed in each different individual).

Remark 3. Cluster analysis focus on the row of the matrix, while all the other methods we'll see are based on the relationship between the columns

Important remark 2. For the moment weassume we're dealing with *numeric variables.*

Mean vector We can associate to the matrix a vector of means called $\overline{\mathbf{x}}$, column vector containing column means of data matrix

$$\underset{p\times 1}{\overline{\mathbf{x}}} = \begin{bmatrix} \overline{x}_1 \\ \vdots \\ \overline{x}_i \\ \vdots \\ \overline{x}_p \end{bmatrix}$$

To obtain it in matrix form by premultipling $\mathbf{X}$ by a row vector of all 1 we obtain the sum of all elements in column, then we divide by n to have the mean and finally we transpose to have a column vector

$$\overline{\mathbf{x}} = \left(\frac{\mathbf{1}_n^T \mathbf{X}}{n}\right)^T = \left(\frac{1}{n}\mathbf{1}_n \mathbf{X}\right)^T = \frac{1}{n}\mathbf{X}^t \mathbf{1}_n$$

Mean centered matrix The mean vector can be used to obtain a new data matrix $\tilde{\mathbf{X}}$, the mean centered data matrix

$$\underset{n\times p}{\tilde{\mathbf{X}}} = \begin{bmatrix} \tilde{x}_{11} & \dots & \tilde{x}_{1i} & \dots \tilde{x}_{1p} \\ \dots & \dots & \dots & \dots \\ \tilde{x}_{j1} & \dots & \tilde{x}_{ji} & \dots \tilde{x}_{jp} \\ \dots & \dots & \dots & \dots \\ \tilde{x}_{n1} & \dots & \tilde{x}_{ni} & \dots \tilde{x}_{np} \end{bmatrix}$$
$$= \begin{bmatrix} x_{11}-\overline{x}_1 & \dots & x_{1i}-\overline{x}_i & \dots & x_{1p}-\overline{x}_p \\ \dots & \dots & \dots & \dots & \dots \\ x_{j1}-\overline{x}_1 & \dots & x_{ji}-\overline{x}_i & \dots & x_{jp}-\overline{x}_p \\ \dots & \dots & \dots & \dots & \dots \\ x_{n1}-\overline{x}_1 & \dots & x_{ni}-\overline{x}_i & \dots & x_{np}-\overline{x}_p \end{bmatrix}$$

The mean vector of this matrix will be the $\mathbf{0}_p$ mean vector.
Which operation to apply to $\mathbf{X}$ and $\overline{\mathbf{x}}$ to obtain $\overline{\tilde{X}}$? The difference between two matrix can be obtained only if we have matrix of the same size; we need to subtract from $\mathbf{X}$ a matrix where each row is a copy of $\overline{\mathbf{x}}$

$$\underset{n\times p}{\overline{\mathbf{X}}} = \underset{n\times 1}{\mathbf{1_n}}\underset{1\times p}{\overline{\mathbf{x}}^{\mathbf{T}}}$$
$$= \begin{bmatrix} \overline{x}_1 & \dots & \overline{x}_i & \dots & \overline{x}_p \\ \dots & \dots & \dots & \dots & \dots \\ \overline{x}_1 & \dots & \overline{x}_i & \dots & \overline{x}_p \\ \dots & \dots & \dots & \dots & \dots \\ \overline{x}_1 & \dots & \overline{x}_i & \dots & \overline{x}_p \end{bmatrix}$$

Therefore

$$\tilde{\mathbf{X}} = \mathbf{X} - \mathbf{1_n}\overline{\mathbf{x}}^{\mathbf{T}} = \mathbf{X} - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\mathbf{X}$$
$$= \underbrace{\left(\mathbf{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right)}_{\mathbf{A}}\mathbf{X}$$

where in the final passage we collected X (to the right) and leave the identity matrix which leaves the product unchanged.
The matrix $\mathbf{A}$ is very interesting from the algebraic pov:

- is usually called the *centering matrix*: if we premultiply a matrix for $\mathbf{A}$ we obtained the mean centered matrix;
- is $n \times n$;
- is symmetric: can be verified that coincides with its transpose ($\mathbf{1}_n\mathbf{1}_n^T$ transposed is the same), but also that the difference between two symmetric matrix is still simmetric ($\mathbf{I}_n$ is simmetric)
- is idempotent: $\mathbf{A} = \mathbf{A}^2$

Standardized data matrix A third matrix of interest is the Standardized data matrix. To standardize a single cell we have

$$z_{ji} = \frac{x_{ji} - \overline{x}_i}{s_i}$$

where the numerator is clearly taken from $\tilde{X}$ while at the denominator we have the standard deviation of i-th column/variable. To compute in matrix form we need to perform matrix operation that given the vector of the standard deviation.
How to divide each column by its standard deviation? We build a diagonal matrix which have the variances s_i^2 on the main diagonal

$$\underset{p \times p}{\mathbf{D}} = \begin{bmatrix} s_1^2 & \dots & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & s_i^2 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & \dots & s_p^2 \end{bmatrix}$$

then we define its power of $-1/2$

$$\underset{p \times p}{\mathbf{D}^{-\frac{1}{2}}} = \begin{bmatrix} \frac{1}{s_1} & \dots & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \frac{1}{s_i} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & \dots & \frac{1}{s_p} \end{bmatrix}$$

Thus we have that

$$\underset{n \times p}{\mathbf{Z}} = \underbrace{\begin{bmatrix} x_{11} - \overline{x}_1 & \dots & x_{1i} - \overline{x}_i & \dots & x_{1p} - \overline{x}_p \\ \dots & \dots & \dots & \dots & \dots \\ x_{j1} - \overline{x}_1 & \dots & x_{ji} - \overline{x}_i & \dots & x_{jp} - \overline{x}_p \\ \dots & \dots & \dots & \dots & \dots \\ x_{n1} - \overline{x}_1 & \dots & x_{ni} - \overline{x}_i & \dots & x_{np} - \overline{x}_p \end{bmatrix}}_{\tilde{\mathbf{X}}} \underbrace{\begin{bmatrix} \frac{1}{s_1} & \dots & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \frac{1}{s_i} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & \dots & \frac{1}{s_p} \end{bmatrix}}_{\mathbf{D}^{-\frac{1}{2}}}$$

And at the same time

$$\underset{n \times p}{\mathbf{Z}} = \tilde{\mathbf{X}}\mathbf{D}^{-\frac{1}{2}} = \mathbf{AXD}^{-\frac{1}{2}}$$

Remark 4. So we can work with raw data, mean centred variables or standardized variables. Which matrix is better to use is something we'll discuss.

Besides deriving this data matrices we can obtain other derivate matrices

Variance-covariance matrix Each variable has its own variance but there's covariance betweem as well.

If we have p variables/variances, actually the number of unique covariances is sum of the first p natural number which is in turn is equivalent to

$$ncovs = \frac{(p-1)cdotp}{2}$$

We want to find the covariance matrix usually denoted by S which is a $p \times p$ matrix that has variances on the main diagonal and covaraince outside

$$\underset{p\times p}{\mathbf{S}} = \begin{bmatrix} s_1^2 & \dots & s_{1i} & \dots & s_{1p} \\ & & s_i^2 & \dots & s_{ip} \\ & & & & s_p^2 \end{bmatrix}$$

It's symmetric so people usually write just the upper triangular part. How to obtain **S** starting from our original data matrix? Let's see the component for a generic element on the diagonal (variance) and out of it (covaraince) respectively

$$\begin{aligned} s_i^2 &= \frac{\sum_{j=1}^{n}(x_{ji} - \overline{x}_i)^2}{n} \\ s_{il} &= \frac{\sum_{j=1}^{n}(x_{jl} - \overline{x}_l)(x_{ji} - \overline{x}_i)}{n} \end{aligned}$$

with s_{il} the covariance between i-th column and l-th column.
The main ingredient are the element of matrix $\tilde{\mathbf{X}}$ of deviations from the mean. **X** is *nxp* thus

$$\underset{p\times p}{\mathbf{S}} = \frac{1}{n}\underset{\tilde{\mathbf{X}}}{\tilde{\mathbf{X}}^T}$$

We have that **S** is

- square p x p
- symmetric (we know from property of covariance) but as well if we transpose above we obtain the same
- i it is positive semidefinite: what does it mean. Several way to determine it:
  - either the eigenvalues are not negative ($>= 0$)
  - the deteminant of the matrix and determinant of all the minors are non negative (we use this way to prove that S is positive semidefinite)

Dimostrazione. To prove **S** is positive semidefinite (in the simple case of just two variables).
Let's consider just two mean-centered variables

$$S = \frac{1}{n}\tilde{\mathbf{X}}^T\tilde{\mathbf{X}} = \frac{1}{n}\begin{bmatrix} dev(x_1) & codev(x_1, x_2) \\ codev(x_1, x_2) & dev(x_2) \end{bmatrix}$$

where $dev(x_1)$ is the deviance (numerator of the variance) while $codev(x_1, x_2)$ is the codeviance (numerator of covariance).
Let's consider the determinant of **S** det **S** and the determinant of its minor (here we have just 1 minor obtained by cancelling the first row and first column, which is the scalar/matrix with just $dev(x_2)$ and its deteminant is just scalar itself

$$\det([dev(x_2)]) = dev(x_2) > 0$$

while for the determinat of $\mathbf{S}$

$$\det \mathbf{S} = dev(x_1) \cdot dev(x_2) - codev^2(x_1, x_2)$$

Is this quantity ≥ 0? if so S is positive semidefinite

$$\begin{aligned} dev(x_1) \cdot dev(x_2) - codev^2(x_1, x_2) \geq 0 \\ dev(x_1) \cdot dev(x_2) \geq codev^2(x_1, x_2) \end{aligned}$$

By dividing both sides by $dev(x_1) \cdot dev(x_2)$ (and reverting order) we have that determinant is ≥ 0 as long as

$$\frac{codev x_1, x_2}{dev x_1 dev x_2} \leq 1 \qquad\qquad r_{12}^2 \leq 1$$

where r_{12}^2 is the R^2, that is the square of correlation coefficient between the two variables r_{12},
By definition the R^2 is always ≤ 1, being the correlation coefficient in the range -1 to 1 So this is true by definition and proves that $\mathbf{S}$ is positive semidefinite □

TODO: if it is positive definite we have if there are CHECK 1:27 di 01

Important remark 3.

Correlation matrix Another matrix we can derive is the correlation matrix $\underset{p\times p}{\mathbf{R}}$ of the p variables:

$$\underset{p\times p}{\mathbf{S}} = \begin{bmatrix} 1 & \dots & r_{1i} & \dots & r_{1p} \\ & & 1 & \dots & r_{ip} \\ & & & & 1 \end{bmatrix}$$

Its generic element is (correlation between i-th variable and l-th variable)

$$r_{il} = \frac{cov(i,l)}{sd(i)sd(l)} = \frac{s_{il}}{s_i s_l} = \frac{\sum_{j=1}^n (x_{ji} - \overline{x}_i)(x_{jl} - \overline{x}_l)}{\sqrt{\frac{\sum_{j=1}^n (x_{ji} - \overline{x}_i)^2}{n} \cdot \frac{\sum_{j=1}^n (x_{jl} - \overline{x}_l)^2}{n}}}$$

How to obtain it?

1.2 Giorno 2

we started seeing how to summarize data and relationship between them
$\tilde{\mathbf{X}}$ is obtained by centering $\mathbf{X}$ using $\mathbf{A}$ with premultiplication
the same happen if we premultiply a vector by $\mathbf{A}$ to center it
standardization is aimed at eliminating the unit of measure of data
$\mathbf{Z}$ is $n \times p$ matrix
the average vector of $\mathbf{Z}$ is still the null vector $\overline{\mathbf{z}} = 0$ (standardized variable have zero mean and 1 standard deviation
covariance matrix $\mathbf{S}$ is square, symmetric (since $cov(x_1, x_2) = cov(x_2, x_1)$

$$\mathbf{S} = \frac{1}{n} \tilde{\mathbf{X}}^T \tilde{\mathbf{X}}$$

where $\tilde{\mathbf{X}}^T\tilde{\mathbf{X}}$ is called **deviance/codeviance matrix** (numerators of variances/covariances)

Here we used the biased version of the matrix (with n at denominator of each element, not $(n-1)$. We could define as well

$$\widehat{\mathbf{S}} = \frac{1}{n-1}\tilde{\mathbf{X}}^T\tilde{\mathbf{X}}$$

which is the **unbiased covariance matrix**

Remark 5. For our purposes using one or the other is the same but if needed to change between the two is done easily by multiplying by n and divide by $n-1$ or viceversa

Important remark 4. Notation: for quantities referring to the population we use greek letters, while roman letters will be used for sample quantities. At the population level:

- the equivalent of S is denoted as $\mathbf{\Sigma}$, which is called **population covariance matrix**
- the equivalent of the mean vector $\overline{x}$ will me is **mu**
- the population analog of R is denoted by $\boldsymbol{\rho}$, the population correlation matrix

Important remark 5. The fact that S cannot have negative eigenvalues have a statistical interpretation since will se that eigenvalue are variances (that cannot be negative).
Negative eigenvalue for $\mathbf{S}$ means something gone wrong in the computation

Correlation matrix is still *pxp*: outside main diagonal linear correlation coefficients between observations (-1 to 1), while on the main diagonal all are 1 (correlation between observation and itselv)

How to compute it? since because the numerator is the generic element of S we need to divide both by std of one and the other variables and in matrix form we pre and postmultiply by $D^{-1/2}$

$$\mathbf{R} = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}$$

Some manipulation highlight the relation between correlation $\mathbf{X}$ and $\mathbf{Z}$

$$\begin{aligned}\mathbf{R} &= \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}\\ &= \mathbf{D}^{-1/2}\underbrace{\frac{1}{n}\tilde{\mathbf{X}}^T\tilde{\mathbf{X}}}_{\mathbf{S}}\mathbf{D}^{-1/2}\\ &= \frac{1}{n}\underbrace{\mathbf{D}^{-1/2}\tilde{\mathbf{X}}^T}_{\mathbf{Z}^T}\underbrace{\tilde{\mathbf{X}}\mathbf{D}^{-1/2}}_{\mathbf{Z}}\\ &= \frac{1}{n}\mathbf{Z}^T\mathbf{Z}\end{aligned}$$

So we can obtain $\mathbf{R}$ from $\mathbf{S}$ or from $\mathbf{Z}$

What is the covariance between z_i and z_l?

$$cov(z_i, z_l) = \frac{1}{n}\sum_{j=1}^{n}(z_{ji} - \underbrace{\overline{z}_i}_{=0})(z_{jl} - \underbrace{\overline{z}_l}_{=0}) = \frac{1}{n}\sum_{j=1}^{n} z_{ji} \cdot z_{jl}$$

So the covaraince of the standardized data matrix is actually

$$cov(\mathbf{Z}) = \frac{1}{n}\mathbf{Z}^T\mathbf{Z} = \mathbf{R}$$

Important remark 6. Either covariance matrix between standardized variables is the correlation matrix or the correlation matrix can be thaught as the covatiance matrix of standardized variables

TODO: check min 22

every time we work with **R** it's like we're working with standardized
Since **R** is a covariance matrix it has the property of any covariance matrix so

- it is pxp simmetric (if we transpose we get the same)
- it is positive semi definite

which are important property for what we'll be doing
if interested in study we can work either

- with S if raw data
- with R if we want get rid of different measurement units

These 5 matrix will be the main ingredients of what we going to do

1.3 Lab

```
# matrix computation
job <- read.table("data/job_perf.txt", header = TRUE)

## see what the data is about
head(job)

##   commun probl_solv logical learn physical appearance
## 1     12         52      20    44       48         16
## 2     12         57      25    45       50         16
## 3     12         54      21    45       50         16
## 4     13         52      21    46       51         17
## 5     14         54      24    46       51         17
## 6     14         48      20    47       51         18

## data about performances: problem solving, logical, learning etc

## now we look at how to compute matrixes

## job is a data.frame: to transform it to matrix
```

```
X <- as.matrix(job)
n <- nrow(X)

## start by centering the data
C <- diag(n) - (1/n) * rep(1, n) %*% t(rep(1, n))
Xc <- C %*% X # centered data

## check it's centered (mean should be 0)
colMeans(Xc)

##        commun    probl_solv      logical         learn      physical
## -5.639933e-16  6.679102e-15 -3.792522e-15 -9.414691e-16  1.059597e-14
##    appearance
## -2.646772e-15

## covariance matrix
S <- 1/n * t(Xc) %*% Xc ## S is 6x6

## differently in cov we get the unbiased covariance
## so to obtain the same
cov(X) * (n-1) / n

##            commun probl_solv logical  learn physical appearance
## commun     7.3776     0.8512  1.5064 7.4896   6.3312     7.7992
## probl_solv 0.8512     5.6944  2.1368 0.9552   0.9344     0.9104
## logical    1.5064     2.1368  6.0596 1.5944   1.5168     1.5788
## learn      7.4896     0.9552  1.5944 7.8816   6.5752     8.0832
## physical   6.3312     0.9344  1.5168 6.5752   5.6944     6.8504
## appearance 7.7992     0.9104  1.5788 8.0832   6.8504     8.7764

## for the standardized data: D is diagonal matrix that has
D <- diag(diag(S))
# diag(S) only extracts the diagonal of our matrix (which has the variances):
# we use another diag to make a diagonal matrix

## do do the ^{-1/2}  we have to do solve which correspond to -1
Z <- Xc %*% solve(D^0.5)

## check that standardized (mean = 1, sd = var = 1)
round(colMeans(Z), 5) # all 0

## [1] 0 0 0 0 0 0

apply(Z, 2, var) # more or less all standardized

## [1] 1.020408 1.020408 1.020408 1.020408 1.020408 1.020408

## correlation matrix: can be found in different ways (covaraince of
## standardized data or starting from the covaraince matrix)
(R <- 1/n * t(Z) %*% Z) #cov: on diagonal 1 (it's a corr)
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 1.0000000 0.1313258 0.2252998 0.9821863 0.9767974 0.9692466
## [2,] 0.1313258 1.0000000 0.3637625 0.1425815 0.1640910 0.1287805
## [3,] 0.2252998 0.3637625 1.0000000 0.2307109 0.2582155 0.2164945
## [4,] 0.9821863 0.1425815 0.2307109 1.0000000 0.9814717 0.9718918
## [5,] 0.9767974 0.1640910 0.2582155 0.9814717 1.0000000 0.9690222
## [6,] 0.9692466 0.1287805 0.2164945 0.9718918 0.9690222 1.0000000
```

```
(R <- solve(D^0.5) %*% S %*% solve(D^0.5)) ## same results obtained with D^{-1/2} S D
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 1.0000000 0.1313258 0.2252998 0.9821863 0.9767974 0.9692466
## [2,] 0.1313258 1.0000000 0.3637625 0.1425815 0.1640910 0.1287805
## [3,] 0.2252998 0.3637625 1.0000000 0.2307109 0.2582155 0.2164945
## [4,] 0.9821863 0.1425815 0.2307109 1.0000000 0.9814717 0.9718918
## [5,] 0.9767974 0.1640910 0.2582155 0.9814717 1.0000000 0.9690222
## [6,] 0.9692466 0.1287805 0.2164945 0.9718918 0.9690222 1.0000000
```

```
cor(X) ## same again
```

```
##               commun probl_solv   logical     learn  physical appearance
## commun     1.0000000  0.1313258 0.2252998 0.9821863 0.9767974  0.9692466
## probl_solv 0.1313258  1.0000000 0.3637625 0.1425815 0.1640910  0.1287805
## logical    0.2252998  0.3637625 1.0000000 0.2307109 0.2582155  0.2164945
## learn      0.9821863  0.1425815 0.2307109 1.0000000 0.9814717  0.9718918
## physical   0.9767974  0.1640910 0.2582155 0.9814717 1.0000000  0.9690222
## appearance 0.9692466  0.1287805 0.2164945 0.9718918 0.9690222  1.0000000
```

Capitolo 2

Cluster analysis

So far we focused on the columns (variances mean covariances). Now we start to see the dataset row-wise, our focus is to measure similarity or dissimilarity of units. I want to read the data matrix row-wise and compare each unit to each other units.

We need to define some metrics and matrices before proceeding. Disclaimer: we have more object that letters in our alphabet sometimes we use the same letter but the context will clarify Here the matrix D will be used as distance matrix (previously used as diagonal matrix)

2.1 Distance matrix

The distance or dissimilarity matrix $\mathbf{D}$ is $n \times n$ and comprises all the distances/dissimilarity between units belonging to the sample. In most cases it is symmetric (if we use distance, which are symmetric) but not need to be so.
Let's consider distance function between units. The various type depends on the kind of data they apply

2.1.1 Distances

2.1.1.1 All numeric variables

How to compare units where all the variable are numerical? With all numeric variables we can think of each unit as a point in the p-dimensional space

Example 2.1.1. In two dimensions

```
         height weight
vadim       182     80
daulet      185     87
```

data can be represented as point in the two dimensional space.
With tree variables we have points in space, with p-variable the unit are point in the p-dimensional variable.

Since data are mapped to points i can measure the difference between units as distance between corresponding points the more different they are the more distant the corresponding will be. while if the points are overlapping the distance will be 0

There are many definition of distances for all quantitative variables

- **euclidean distance**: in the 2-dimensional space is the length of the segment joining the two points (just applying pitagora's theorem). n general the distance between unit j and h is

  $$d_{jh} = \sqrt{\sum_{i=1}^{p}(x_{ji} - x_{hi})^2}$$

  which is also called the L_2-norm

- city-block/Manhattan distance:

  $$d_{jh} = \sum_{i=1}^{p} |x_{ji} - x_{hi}|$$

  The name comes from the fact that in Manhattan we can't walk diagonal so we need to count the segments of streets which separates two point. Also called L_1-norm

- **Minkowski distance**: the previous are specialization of the following family of distances

  $$d_{jh} = \sqrt[m]{\sum_{i=1}^{p} |x_{ji} - x_{hi}|^m}$$

  If $m = 1$ we obtain the city block distance, while if $m = 2$ we obtain the Euclidean distance.
  In general larger and values for m put larger weights on large differences (difference is large and raise to a large power take power on the other differences on other variables).

Example 2.1.2. To see the impact of increasing m on variables relevance see

$$\begin{aligned}
&\text{manhattan:} \quad d_{vd} = |182 - 185| + |80 - 87| = 3 + 7 = 10 \\
&\text{euclidean:} \quad d_{vd} = \sqrt{(182 - 185)^2 + (80 - 87)^2} = \sqrt{9 + 49} = \sqrt{58} = 7.62
\end{aligned}$$

Going from Manhattan to Euclidean distance become smaller but the importance of weight variable (were the difference between units is higher) become bigger

2.1.1.2 All binary variables

Example 2.1.3. Suppose

```
        brother  smoke  likebeer likechocolate
vadim         1      0         1             0
daulet        1      0         0             1
```

V D	1	0	
1	a	b	
0	c	d	
			p

Tabella 2.1: The mf counts table

How to measure how units are similar with respect to p binary variables. A table such as 2.1 is constructed with the count of variable where the two units have agreement or disagreement where

- a: number of times (variables) where units agree on 1
- d: number of times the units agree on 0
- b: n of times unit i shows the 1 while the unit h show 0 (disagree)
- c: n of times unit i shows the 0 while the unit h show 1 (disagree)

All sum of all the entries will be p, the number of variable. In the example above we have $a = b = c = d = 1$. Basing ourselves on the matrix we have several dissimilarity coefficients:

1. *simple dissimilarity coefficient*
$$\begin{aligned} d_{ih} &= \frac{\text{n. disagree}}{n.\ variables} = \frac{b+c}{a+b+c+d} \\ &= 1 - s_{ih} = 1 - \frac{a+d}{a+b+c+d} \end{aligned}$$
where s_{ih} is called simple similarity coefficient.
It's called *dissimilarity* and not *distance* because a distance function requires to be positive symmetric AND to satisfy triangular inequality; this function does not satisfy the third property (dissimilarity usually don't)

2. *Jaccard coefficient*: is defined as
$$d_{ih} = \frac{a}{a+b+c}$$
These measures were defined for taxonomy studies and the used to compare different animals. In these cases there are many agreement on not having certain characteristics. But having agreement on not having wings (cows and snake) does not necessarily mean that two animals (units) are similar.
Generalizing Jaccard consider in many cases the agreement on null non necessarily means similarity. Jaccard invented the jaccard coefficient, which eliminates in the numerator and denominator the agreement on null

Example 2.1.4. In our example, with $a = b = c = d$ we have that the simple dissimilarity between vadim and daulet is

$$d_{vd} = \frac{2}{4} = 0.5$$

so the two units are identical with respect to half o characteristics considered.

2.1.1.3 All categorical variables

Example 2.1.5. Suppose

```
        phone   fav.team  zodiac blood
vadim  iphone barcellona  gemini    A-
daulet iphone juventus      lion    B+
```

Difference between the units is just the number of agreement/number of variable

$$d_{jh} = \frac{\text{n.agreement}}{\text{n.variables}} = \frac{\text{n.agreement}}{p}$$

However this is a poor measure of similarity which does not take account the number of levels of each variables (eg is easy to have distance in zodiac sign more easy in the phone iphone vs android)

2.1.1.4 Mixed types

Example 2.1.6.

```
          height  weight   phone brother beer
 vadim    182       80  iphone       1    1
daulet    185       87  iphone       1    0
```

A popular measure here is the Gower's dissimilarity coefficient which is defined as the complement of similarity measure

$$d_{jh} = 1 - s_{jh}$$
$$s_{jh} = \frac{\sum_{i=1}^{p} s_{jhi} \cdot \delta_{jhi}}{\sum_{i=1}^{p} \delta_{jhi}}$$

Regarding

- δ_{jhi} which is an indicator defined as

  $$\delta_{jhi} = \begin{cases} 1 & \text{if comparison between unit } j \text{ and } h \text{ is possible for variable } i \\ 0 & \text{otherwise} \end{cases}$$

  This indicator take into account the very likely situation that for 1 unit a piece of information is lacking: if we have missing values the two units cannot be compared and in this case it is impossible to use data to tell difference. If all the variables can be compared at the numerator we have p, which is a counter where both the units are non-missing.
  δ_{jhi} is a simply way to taking into account missingness

- the similarity s_{jhi} quantity is defined separatedly for *each variable*. If the variables are

  - numeric

    $$s_{jhi} = \frac{1 - |x_{ji} - x_{hi}|}{\text{range of } x_i} = 1 - \frac{x_{ji} - x_{hi}}{max(x_i) - min(x_i)}$$

– binary

$$s_{jhi} = \begin{cases} 1 & \text{the units show the same level} \\ 0 & \text{otherwise} \end{cases}$$

Example 2.1.7. Eg for height which is numerical

$$s_{vd,height} = 1 - \frac{|182 - 185|}{190 - 160} = 1 - \frac{3}{30} = \frac{9}{10}$$

where 190 and 160 were the height of taller and smaller in class. So being 9/10 the two units are very similar (close to 1).
In case of considering the similarity of the most distant units we have

$$s_{vd,height} = 1 - \frac{|190 - 160|}{190 - 160} = 0$$

and we have max difference.
For what concerns the similarity in weight

$$s_{vd,weight} = 1 - \frac{|87 - 80|}{90 - 48} = 1 - \frac{7}{46} = \frac{5}{6} = 0.83$$

where 90 and 48 are the weights of the fatter and thinner in class. So 0.83 is still highly similar units.
Thus finally overall we have that the Gower dissimilarity is

$$d_{dv} = 1 - s_{dv} = 1 - \frac{0.9 + 0.83 + 1 + 0}{4} = 1 - 0.68 = 0.32$$

Dissimilarity between the two units is 0.32 and we do the same things for all the units

2.2 Clustering

A *cluster* is a group of unit.

Goal of this analysis to find groups of units that are similar within group and separate/different between as much as we can (wrt the variable we take into account).

Important *starting point* in variable selection: we have to choose carefully the charateristics to analyze (same set of units can provide very different information).

Cluster analysis is composed by a very broad sets of techniques: one can define groups in many way and thus obtain many different solutions.

The most traditional way cluster analysis is performed we have two main method:

- *hierarchical methods*: the methods work by building a series of nested classification (small group inside larger and so on). we don't have a single clustering
- partitioning methods:

Hierarchical methods can be divided in two family (based on the way on which is builded):

- *agglomerative methods* (bottom-up) (which is the one we will study): start from n cluster group each composed by 1 unitm, and we put clusters together/merge them until we obtain the big group. The procedure/merging process is depicted by a dendrogram

- *divisive methods* (top-down): splitting in subgroups up to when each group has 1 unit.
  Most famous divisive methods is due to Edwards and Cavalli Sforza (biological domani, numerical variables only based onthe decomposition of total sum of square in the within/between, in order to have the smallest within and larger between). One goes on up to when variability is zero in each group ($n = 1$ for each).
  This approach is not very much used because very computational demanding (we have to try each splitting) and furthermore only for numerical variabile. Finally if we make mistakes in the first stages, the error goes on and on (contrary to agglomerative)

A group/cluster of just 1 unit is called *singleton*.

2.3 Hierarchical agglomerative methods

We know how to compute the dissimilarity between pair of units; now we need a measure *distance between groups* (to make them as distant possible): according on how to define the similarity of groups we have different clustering solutions. For group dissimilarity we start from pairwise similarity and have several approach:

1. *single linkage*: minimum distance between units belonging to different cluster (least distant couple)

2. *complete linkage*: maximum distance between units belonging to different cluster (most distant couple of units)

3. agerage linkage: mean distance between units belonging to different cluster (compromise of the two above)

Remark 6. The previous methods works for *any variable* (eg we can apply Jaccard, Gower and so on to calculate the distances between units/groups).

Among the agglomerative methods there are two more options that works for *numerical variables only*:

1. centroid methods

2. Ward's methods

Important remark 7. In case in, if in the distance matrix we have two *identical minimum distances*, we toss a coin to choose which one to aggregate.

2.3.1 Single linkage

Having two groups C and G their distance is the minimum distance between the units

$$d_{CG} = \min_{j \in C, h \in G} d_{jh}$$

single linkage is also known as *nearest neighbor*. Thus one has to compute all the pairwise distances between all the units in the first and the second group: then the distance between the groups is the distance between the closest point belonging to the different groups.

Example 2.3.1. Assume we have thid distance matrix D between units of our sample (a, b, c, e) per il momento indico in questo modo la matrice facendo riferimento alle unità ...

$$\mathbf{D} = \begin{bmatrix} & a & b & c & e \\ a & 0 & 1 & 4 & 3 \\ b & & 0 & 2 & 7 \\ c & & & 0 & 5 \\ e & & & & 0 \end{bmatrix}$$

To cluster this dataset using single link we start by putting together the closest units: a and b are the two most similar which have **distance 1**. To update the distance matrix and re-iterate, we apply the single linkage definition to calculate the distance of these groups. We obtain

$$\mathbf{D} = \begin{bmatrix} & (a,b) & c & e \\ (a,b) & 0 & 2 & 3 \\ c & & 0 & 5 \\ e & & & 0 \end{bmatrix}$$

where the elements of the new first row (the only changing) were calculated as

$$d_{(a,b),c} = \min(d_{ac}, d_{bc}) = \min(4, 2) = 2$$
$$d_{(a,b),e} = \min(d_{ae}, d_{be}) = \min(3, 7) = 3$$

Now we iterate and choose to aggregate the group with the minimum distance which are (a, b) and c with **distance 2**. The new distance will be

$$\mathbf{D} = \begin{bmatrix} & (a,b,c) & e \\ (a,b,c) & 0 & 3 \\ e & & 0 \end{bmatrix}$$

where we obtained

$$d_{(a,b,c),e} = \min(d_{ae}, d_{be}, d_{ce}) = \min(3, 5, 7) = 3$$

So the final distance between the units will be **3**.
To build the dendogram, we put units on the x axis and distances on which are grouped on the y axis

TODO: figura dendogramma lezione 3, primo esempio

This is clustering obtained with single linkage: risk of this method is to obtain strange groups since has the tendency to put togheter heterogenous units. This because single linkage is affected by *chaining effect*: assume that we have two groups because the near two YELLOW ighlighet points the two groups tend to be put togheter (in subsequent iterations)

TODO: seconda immagine

It has the advantage of being flexible

2.3.2 Complete linkage

Also called *farthest neighbors*, the distance between groups is defined as

$$d_{CM} = \max_{j \in C, h \in G} d_{jh}$$

Example 2.3.2. Starting from the very same matrix of differences

$$\mathbf{D} = \begin{bmatrix} & a & b & c & e \\ a & 0 & 1 & 4 & 3 \\ b & & 0 & 2 & 7 \\ c & & & 0 & 5 \\ e & & & & 0 \end{bmatrix}$$

we put together the most similar units, so a, b (which have **distance 1**); the only change is in how we calculate the distance between groups and update the distance matrix

$$\mathbf{D} = \begin{bmatrix} & (a,b) & c & e \\ (a,b) & 0 & 4 & 7 \\ c & & 0 & 5 \\ e & & & 0 \end{bmatrix}$$

where we calculated

$$d_{(a,b),c} = \max d_{ac}, d_{bc} = \max 4, 2 = 4$$
$$d_{(a,b),e} = \max d_{ae}, d_{be} = \max 3, 7 = 7$$

Now to aggregate I look for the smallest value, which as before (not always the case, btw) are between (a, b) and c with **distance 4**. We construct the final matrix as

$$\mathbf{D} = \begin{bmatrix} & (a,b,c) & e \\ (a,b,c) & 0 & 7 \\ e & & 0 \end{bmatrix}$$

where

$$d_{(a,b,c),e} = \max d_{ae}, d_{be}, d_{ce} = \max 3, 5, 7 = 7$$

So finally we put all the units together with **distance 7**.

TODO: dendogram

Complete linkage tends to produce *spherical grops*: it splits data into balls when we have big jump it means that we're trying to put together groups which are more different

2.3.3 Average linkage

The distance between groups is calculated as:

$$d_{CG} = \frac{\sum_{j \in C, h \in G} d_{jh}}{n_c \cdot n_g}$$

where at the denominator we have all the comparison between units of the two groups.

Example 2.3.3. Starting from the very same matrix

$$\mathbf{D} = \begin{bmatrix} & a & b & c & e \\ a & 0 & 1 & 4 & 3 \\ b & & 0 & 2 & 7 \\ c & & & 0 & 5 \\ e & & & & 0 \end{bmatrix}$$

After the first aggregation

$$\mathbf{D} = \begin{bmatrix} & (a,b) & c & e \\ (a,b) & 0 & 3 & 5 \\ c & & 0 & 5 \\ e & & & 0 \end{bmatrix}$$

where we obtained

$$d_{(a,b),c} = \frac{1}{2 \cdot 1}(4+2) = 3$$
$$d_{(a,b),e} = \frac{1}{2 \cdot 1}(3+7) = 5$$

The second step is to merge unit c with group (a,b) (having **distance 3**) obtaining

$$\mathbf{D} = \begin{bmatrix} & (a,b,c) & e \\ (a,b,c) & 0 & 5 \\ e & & 0 \end{bmatrix}$$

where

$$d_{(a,cb,c),e} = \frac{1}{3 \cdot 1}(3+7+5) = 5$$

The final one put together the remaining, having distance 5

TODO: dendrogram

2.3.4 Centroid methods

This and the following methods are for numeric variables only.

$$d_{CG} = \|\overline{\mathbf{x}_C} - \overline{\mathbf{x}_G}\|$$

where $\overline{\mathbf{x}_C}$ is the mean vector of c and $\overline{\mathbf{x}_G}$ is the mean vector of G, That is these are the mid points of the cloud of points associated to a given group in the space. and the distance between them is the distance between the cluster they represent.

Remark 7. Obviously we need numeric variables only to compute the mean vectors; in principle we would need to compute Lance willians formula linkage method
two statistic ian put all the clustering methods in a general family and changing parameter gives us. This is way the centroid put together

TODO: riascolterei qui

2.3.5 Ward's method

Still included in Lance-Williamms, this is the agglomerative analog of Edward-Cavalli-Sforza divisive method. It's based on the decomposition of total sum of square in the within and between components

- We start assuming n cluster each composed by 1 unit: thus our within group sum of WSS=0 (single unit identical to themselves)
- The merging units to groups causes an increase in WSS: so at each step we perform the merge that causes the smallest increase in WSS (create groups which are most similar)

Some remarks:

- since we work with sum of square it's obvious that we need numerical variables
- this was the first method proposed in statistical literature
- has have no strong math properties but they're very populare.

the drawbacks is that

- the cluser they find is localy obtimal (stepwise): if one cut the dendogram at one point and find 5 groups, is not necessarily the best partition overall. it's the best given the steps performed before, all the below in terms of dendrogram, but not necessarily minimize the WSS
- can be slow

TODO: Riascolta $X^T X$ $(m + 1m + 1)$ rank

2.4 Partitioning methods (k-means)

The aim is to provide a single-step partition in k-groups: it's not an hierarchy anymore, its a flat partition.
The most famous methods is the k-means clustering (so as the name says the methods *works for numerics variable only*).

2.4.1 k-means clustering

We want to obtain the partition in k groups, with k given, having the smallest WSS; having our dataset and considered a given k eg $k = 3$. One of the methods (not the unique) with which to implement k-means is the following

```
|----------------------| x <- consider first three rows
|----------------------| x <- of our dataset
|----------------------| x <- as starting centres
|----------------------|
...
|----------------------|
|----------------------|
```

The algorithm start choosing the first three rows in the dataset and consider them as centers of the $k = 3$ groups we want. Then

- i start reading my data matrix and assign the fourth unit to the center to which it is closer. eg i assign unit 4 to the second center
- i compute the center again and I continue reading the matrix in the same way, recomputing the centres after each assignment
- once processed all the dataset and having computed three centers, we start reading again the dataset and reassign all the observation to the nearest center of the three

This method is also known as dynamic cloud because every time a unit is added the cloud, its cloud/means moves (and the centre as well)

So:

- The method we've seen depends on the first three units, for this reason we make a second passage/run in the dataset. The first is to find reasonable centers, the second is to find the groups starting from reasonable centers/to produce the final assingment.
- seems time consuming but it's faster than hierarchical methods. (procedure is called fast clust in SAS) and allows to analyze a large number of units at the same time

Remark 8. Problem with k-means is that one has to chose k in advance: we have to have a criterion how many groups to choose,

2.4.2 Other methods like k-means

Before continuing with k-means we briefly mention two methods which are becoming more and more popular in clustering literature.

2.4.2.1 PAM

Very popular method and close to k-means, PAM[1] stands Partitioning Around Medoids: what a medoid is? it's a prototypical unit. It is an observed unit: as opposite to k-means (where the means doenst correspond to a real object) the medoid is a real observation which act as center. The goal is to find the unit that best represents the groups in the unit; the medoid can be taken as example idea close to k-means but the center is a real unit not a mean of observations algorithmically it becomes more complicate than k-means. This allow to work with non numeric variables we use any distance that we'veseen and find the unit which minimise that distances. Same strategy as k-means as it is a partitioning method but the group centers must be observed units. It can be used with any kind of data while k-means assumes numeric variables only

[1]Peter Rousseaw has invented this method and created the Rousseaw prize (nobel prize for statistics) First winner were research harvard on causal inference. Second those the FALSE discovery rates (benjamini hochberg)

2.4.2.2 Model based clustering

All methods seen so far are distribution free: we made no assumption regarding probability distribution that generated the data.
On the other hand if we can rely on probabilistic assumptions there's a big family of method, the *model based clustering*, which rely on the hypothesis that data are generated by a probabilistic model.
Idea is not new (Pearson did it in univariate case): assume we want study height, so X is height $F(x)$ is pdf of height. After having obtained and seen the data we can imagine that height of the group is generated by sampling from a probability density.
But the point is that in this group there're males and females; so overall F probability seen is a mixture of two probability functions/density, from males and density of females.

$$F(x) = \pi_m F_m(x) + \pi_f F_f(x)$$

π_m and π_f are proportion of males and females proportions so it is assume to be $\pi_m + \pi_f = 1$ in the population. π_m, π_f are known as *mixing proportion*.
Now I can make assumption

$$F_m(X) \sim N(mu_m, sigma_m^2)$$
$$F_f(X) \sim N(mu_f, sigma_f^2)$$

I can rewrite my $F(x)$ as

$$F(x) = \pi_m \phi(x, mu_m, sigma_m^2) + \pi_f \phi(x, mu_m, sigma_m^2)$$

where ϕ is probabilty density function of normal distribution computed in x with mean and variace coming from the two population.
This above is called a *finite mixture model*: in order to obtain the observed and unknown $F(x)$ we need to work on the component μ_m, μ_f (while π_m and π_f are still the mixting proportion). We have to guess the πs and μs.
Pearson started from a simpler case: I *know* the mixing proportions πs, and assume that the shape of my component densities is Gaussian. My problem is to estimate the parameters of the normal distribution which gives me the $F(x)$ that best fit my data. Starting from this idea people have tried to complicate the problem:

- first assuming we don't know the mixing proportion. This can be overcome with different attempt using different mixing proportions
- not knowing the number of cluster
- extending to multiple variables: here we need to estimate mean vector and covariance matrices as well (not single mean and variances). It took a while but recently people invented methods to estimate EM algorithm

TODO: CHECK this 17/2

$$F(\underline{\mathbf{x}}) = \sum_{g=1}^{G} \pi_g \cdot F_g(\underline{\mathbf{x}})$$

with priori probability of group membership $\sum_{g=1}^{G} pi_g = 1$ and F_g groups specific density functions. In the case of gaussian components

$$F(\underline{\mathbf{x}}) = \sum_{g=1}^{G} \pi_g \cdot \phi(\underline{\mathbf{x}}, \mu_g, \Sigma_g)$$

and this is called gaussian mixture models (GMM).
A priori we dont' have number of groups but we have the likelihood to guide us to estimate to select good value for parametrs.
k-means has been proved to be one particular case of these model where variables are uncorrelated and have the same variance (variance covariance matrix is diagonal).
Other methods have been proposed which are not based on gaussian (mixture of multivariate T, and so on).

2.4.3 Choice of k in k-means

Back to k-means, problem with it is that we need to specify k in advance but in many application we don't know the number of groups. So we need criteria:

- what's the most reasonable value for k
- i've partitioned into k groups is good or bad? $k + 1$ is better?

we need to measure quality of clustering: a couple of idea/two different measures

2.4.3.1 Silhouette score

Average silhouette width (by Peter Rousseau, quello del nobel per la statistica), once obtained the clustering, is based on two ingredients:

- $a(j)$ is the average dissimilarity between unit j and the other units onging to the same group
- $b(j)$ average dissimilarity between unit j and the units that belong to the closest group to the one unit j belongs (so I calculate the distabnces between the unit j and all the otgher units in all the other and choose the best group other than that where the unit is assigned)

Then i can calculate the *silhouette of a unit*, defined as **TODO**: CHECK 17/2

$$s(j) = \frac{b(j) - a(j)}{\max[a(j), b(j)]}$$

where we compare the distance of units to that of the most near other group.
I expect that the numerator is positive (more similar to the groups it belongs to respect to other), then I normalize the quantity by considering the max value.
This quantity can go from -1 to +1:

- +1 means that units have been perfectly assigned
- 0 means taht it could be assigned equally to the group who belong or the group with more similar units (in some sense assignment done by clustering is neutral, one group or the other is teh same)

- -1 means wrong sasignment: the unit is more similar to the other group respect to the group who were assigned. the second best group wold have been better

TODO: graph

Example 2.4.1. In the example the silhouette of x_1

$$a(x_1) = \frac{3+5}{2} = 4$$
$$b(x_1) = \min\left(\frac{6+8}{2}, \frac{10+12}{2}\right) = 7$$
$$s(x_1) = \frac{7-4}{7} = \frac{3}{7}$$

So if $s(j)$ is larger than 0, as in this case, than this is good assignment.

Finally to measure the quality of clustering we compute the average silhouette

$$\bar{s} = \frac{\sum_{j=1}^{n} s(j)}{n}$$

some units will be correclty classified, some neutral some badly classified so we expect that the clustering is better if **s** is near to 1. So in general the higher the mean better and good clustering quality is obtained when $\bar{s}$ is close to 1.
Drawback of this approach: here we have to have at least two groups, so this method does not allow to check if actually the best k is 1, that is it doesn't allow to check the actual existence of different groups.
Finally silhouette are plotted in the silhouette plot: graph of the silhouette profile of each individual is reported and coloring according to the group (meaningful if we have few units)

Pearson's gamma

Another measure of clustering quality is known as Pearson's γ: this is nothing but a linear correlation coefficient r, computed on special things

All the cluster we've considered put together a,b,c and let e alone. To compute Pearson's gamma (single linkage example) starting from the original dissimilarity matrix TODOHERE

$$\mathbf{D} = \begin{bmatrix} & a & b & c & e \\ a & 0 & 1 & 4 & 3 \\ b & & 0 & 2 & 7 \\ c & & & 0 & 5 \\ e & & & & 0 \end{bmatrix}$$

First we "vectorize" it putting all upper triangular distances (outside diagonal) in vector form and add a dummy vector/variable which takes value 1 if units are assigned to different group after clustering, 0 otherwise

$$d = \begin{bmatrix} d_{ab} \\ d_{ac} \\ d_{ae} \\ d_{bc} \\ d_{be} \\ d_{ce} \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ 3 \\ 2 \\ 7 \\ 5 \end{bmatrix}, \quad g = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

In the final clustering we had we had a, b, c and e alone; above in g is reported the corresponding vector.

Pearson's γ is just the correlation coefficient between d and g (between the distance and the fact that units are assigned to different groups or the same group

$$\gamma = frac \sum_{i=1}^{n(n-1)/2} d_i g_i - \frac{n(n-1)}{2}\bar{d}\bar{g} \sqrt{\sum_{i=1}^{n(n-1)/2} d_i^2 - \frac{n(n-1)}{2}\bar{d}^2} \sqrt{\sum_{i=1}^{n(n-1)/2} g_i^2 - \frac{n(n-1)}{2}\bar{g}^2}$$
$$= \frac{codev(d,f)}{\sqrt{dev(d)dev(g)}}$$

the sum is on i which are the number of unique elements out of the diagonal. The measure is a correlation which take range between -1 and 1:

TODO: CHECK 17/2

- large positive values close to $+1$ means that distant units tend to be clustered in different group and close units tend to be clustered together
- values close to 0 means that the classification is almost random
- negative values mean that the classification is a mess: we have put in the same group different units and put in different groups similar units.

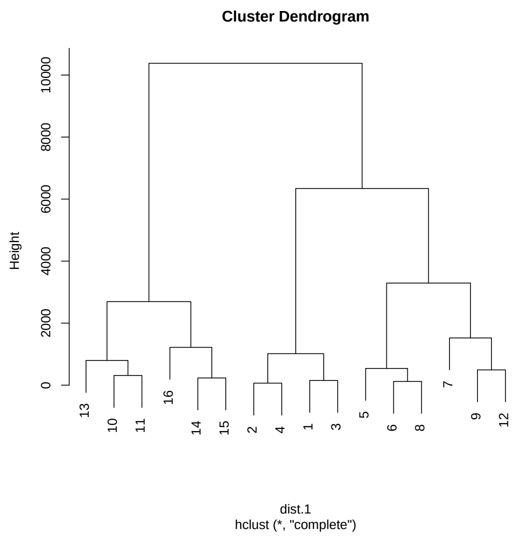
So again as before the larger the better: good clustering correspond to large positive values for γ.

2.5 Lab

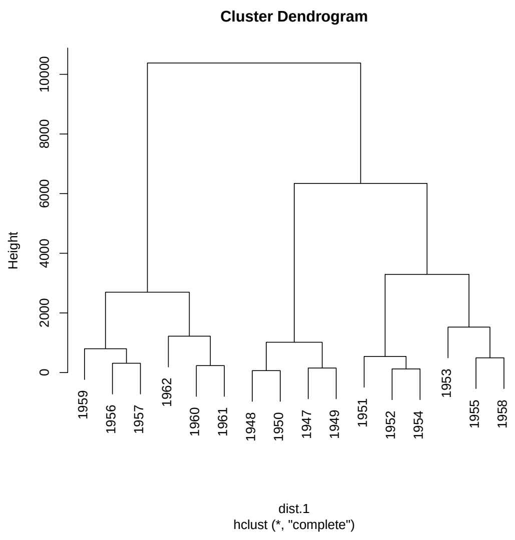
```
## we start by finding distance matrixes among observation
(data <- read.table("data/data.txt", header = TRUE))

##   Att.1 Att.2 Att.3 Att.4 Att.5 Att.6
## 1     1     1     1     0     0     1
## 2     1     0     0     1     0     1
## 3     1     0     0     1     0     1
## 4     0     0     0     0     1     0

## dataset with just binary variables (attributes present or not)
## in situation like this a common distance is jaccard: we compute it using
## dist function (euclidean distance done by default)
dist(data, method = "binary") # it computes lower triangular (symmetric)

##     1   2   3
## 2 0.6
## 3 0.6 0.0
## 4 1.0 1.0 1.0
```

```
# mixed-type data:
library(cluster)
data(flower)
head(flower) # ?flower:

##   V1 V2 V3 V4 V5 V6  V7 V8
## 1  0  1  1  4  3 15  25 15
## 2  1  0  0  2  1  3 150 50
## 3  0  1  0  3  3  1 150 50
## 4  0  0  1  4  2 16 125 50
## 5  0  1  0  5  2  2  20 15
## 6  0  1  0  4  3 12  50 40

# first three binary (third is asimmetric binary), fourth/fifth categorical,
# 6th/7th quantitative

## with this kind of data, the gower distance: in R the function implementing
## it is called daisy: see ?daisy

## we tell daisy the third
gower <- daisy(flower,
               metric = 'gower',
               type = list(asymm = 3)) # treat third is asymmetric binary

## better to check
summary(gower)

## 153 dissimilarities, summarized :
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1418  0.4164  0.5101  0.5098  0.6051  0.8875
## Metric :  mixed ;  Types = N, N, A, N, O, O, I, I
## Number of objects : 18

## we have
## two N: nominal,
## one A: asymmetric
## two O: ordered factor,
## two I: numeric,

## back to dichotomic data let's see the longley dataset
# install.packages("AER")
library(AER)

## Caricamento del pacchetto richiesto:  car
## Caricamento del pacchetto richiesto:  carData
##
## Caricamento pacchetto:  'car'
## Il seguente oggetto è mascherato da 'package:lbmisc':
##
##     recode
## Caricamento del pacchetto richiesto:  lmtest
## Caricamento del pacchetto richiesto:  zoo
##
## Caricamento pacchetto:  'zoo'
## I seguenti oggetti sono mascherati da 'package:base':
##
##     as.Date, as.Date.numeric
## Caricamento del pacchetto richiesto:  sandwich
## Caricamento del pacchetto richiesto:  survival
```

```
data(Longley) # ?Longley: time series, observation are years
## we want to look employment: we need to extract it from the ts
longley <- as.data.frame(Longley)
X <- longley$employment  ## employment is numeric (for regarding distances)
labels.X <- as.character(1947:1962)     ## name of the years

## now find the distance: different distance can be used (most used is euclidean)
dist.1 <- dist(X, method = 'euclidean')
dist.2 <- dist(X, method = 'manhattan')

## we start clustering with hierarchical methods: ?hclust takes as input the
## dissimilarity and then we specify the method

h <- hclust(dist.1) # "complete" linkage (by default) on euclidian distance h
plot(h) # the dendrogram, without rownames it's a mess
```



```
plot(h, labels = labels.X) # much better
```



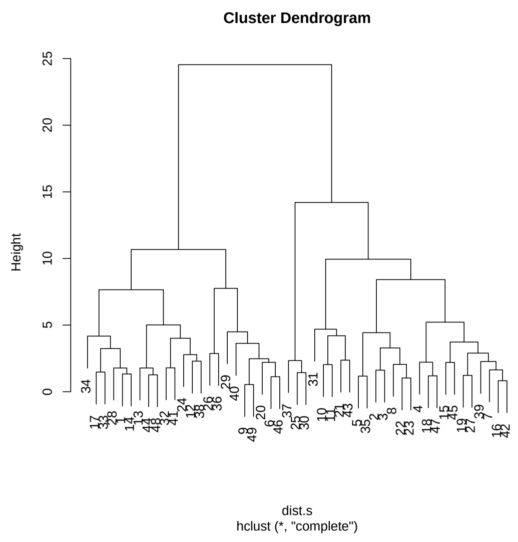
```
## how the tree was built during time?
h$merge

##       [,1] [,2]
##  [1,]   -2   -4
##  [2,]   -6   -8
##  [3,]   -1   -3
##  [4,]  -14  -15
##  [5,]  -10  -11
##  [6,]   -9  -12
##  [7,]   -5    2
##  [8,]  -13    5
##  [9,]    1    3
## [10,]  -16    4
## [11,]   -7    6
## [12,]    8   10
## [13,]    7   11
## [14,]    9   13
## [15,]   12   14

## a matrix with two columns:
## to understand this look at the first with id
## 1) starting from the first row: the dendrogram is built bottom to top
## when we have two negative values (-2 and -4) the observation are merged
## together
## 2) in row 7 a negative and a positive value: the observation with negative
## sign has been merged with cluster built at step said by column: so unit 5
## was ## merged with cluster at row 2, so with observation 6 and 8
## 3) both positive values: 1 and 3 in row 9. Cluster formed in row 1 (2 and 4)
## has been merged with row 3 (observation 1 and 3)

## height is important: look at distances between
```

```
h$height

##  [1]    65   122   152   233   312   494   540   798  1016  1220  1524  2694
## [13]  3292  6342 10380

h.cl = c() # empty vector
h.cl[1] = h$height[1] # difference between height at the second step and first
for (i in 2:length(h$height)){
  h.cl[i] = h$height[i] - h$height[i-1]
}

## h.cl now contains differences between useful to cut the tree: we cut at the
## max distance which.max(h.cl) # the largest difference was obtained in the
## last step

## now we need to extract the height to which it correspond the maximum
## distance h.max <- h$height[ which.max(h.cl) -1 ] # we want to cut the tree
## before the max distance so -1 is needed

## final
plot(h, labels = labels.X)
abline(h = h.max, col = "red")

## Error:  oggetto 'h.max' non trovato

## with complete linkage the best number of group is two
## we extract the classification with cutree

cl <- cutree(tree = h, k = 2) # cut to create two groups
cl

##  [1] 1 1 1 1 1 1 1 1 1 2 2 1 2 2 2 2

cl <- cutree(tree = h, h = h.max) # cut using an height: the h.max

## Error:  oggetto 'h.max' non trovato

## this was complete linkage: other methods one has only to change methods at
## the beginning.

h <- hclust(dist.1, method = "average")

## and then all the code is equal (results may be different) with average
## linkage the best number of group is still 2
```

I had problems with pc, look at scripts uploaded (al back to normal with k-means)

```
## another dataset

head(sparrow <- read.table("data/sparrows.dat", header = TRUE))
```

```
##   totL AlarE  bhL   hL   kL
## 1  156   245 31.6 18.5 20.5
## 2  154   240 30.4 17.9 19.6
## 3  153   240 31.0 18.4 20.6
## 4  153   236 30.9 17.7 20.2
## 5  155   243 31.5 18.6 20.3
## 6  163   247 32.0 19.0 20.9

## dataset about sparrow during a storm: some of them died while some other
## survived (we don't know if they died or survived in the dataset): the died
## are from row 22 to 49 btw

## we want to see if clustering these observation we found a link between their
## death and characteristics
## again all numeric

dist.s = dist(sparrow, method = 'euclidean')

h <- hclust(dist.s) # "complete" linkage (by default) on euclidian distance h
plot(h) # the dendrogram, without rownames it's a mess
```



```
## how the tree was built during time?
h$merge # a matrix with two columns:

##       [,1] [,2]
##  [1,]   -9  -49
##  [2,]  -16  -42
##  [3,]  -22  -23
##  [4,]   -6  -46
##  [5,]   -5  -35
##  [6,]  -18  -47
##  [7,]  -19  -27
##  [8,]  -44  -48
##  [9,]   -1  -14
## [10,]  -25  -30
## [11,]  -17  -33
## [12,]   -2   -3
## [13,]   -7    2
## [14,]  -28    9
## [15,]  -13    8
## [16,]  -32  -41
## [17,]  -10  -11
## [18,]   -8    3
## [19,]  -15  -45
## [20,]  -20    4
## [21,]   -4    6
## [22,]  -39   13
## [23,]  -12  -38
## [24,]  -37   10
## [25,]  -21  -43
## [26,]    1   20
## [27,]  -24   23
## [28,]  -26  -36
## [29,]    7   22
## [30,]   11   14
## [31,]   12   18
## [32,]  -40   26
## [33,]   19   29
## [34,]   16   27
## [35,]  -34   30
## [36,]   17   25
## [37,]    5   31
## [38,]  -29   32
## [39,]  -31   36
## [40,]   15   34
## [41,]   21   33
## [42,]   35   40
## [43,]   28   38
## [44,]   37   41
## [45,]   39   44
## [46,]   42   43
## [47,]   24   45
## [48,]   46   47
## to understand this look at the first with id

## 1) starting from the first row: the dendrogram is built bottom to top when
## we have two negative values (-2 and -4) the observation are merged together

## 2) in row 7 a negative and apositive value: the observation with negative
## sign has been merged with cluster built at step said by column: so unit 5
## was merged with cluster at row 2, so with observation 6 and 8
## 3) both positive values: 1 and 3 in row 9. Cluster formed in row 1 (2 and 4)
## ## has been merged with row 3 (observation 1 and 3)

## height is important: look at distances between h$height
h.cl = c() # empty vector
h.cl[1] = h$height[1] # difference between height at the second step and first
for (i in 2:length(h$height)){
  h.cl[i] = h$height[i] - h$height[i-1]
}

## h.cl now contains differences between useful to cut the tree: we cut at the
## max distance
which.max(h.cl) # the largest difference was obtained in the last step

## [1] 48

## now we need to extract the height to which it correspond the maximum
## distance
h.max <- h$height[ which.max(h.cl) -1 ] # we want to cut the tree before the
                                        # max distance so -1 is needed

## final
plot(h, labels = labels.X)
```

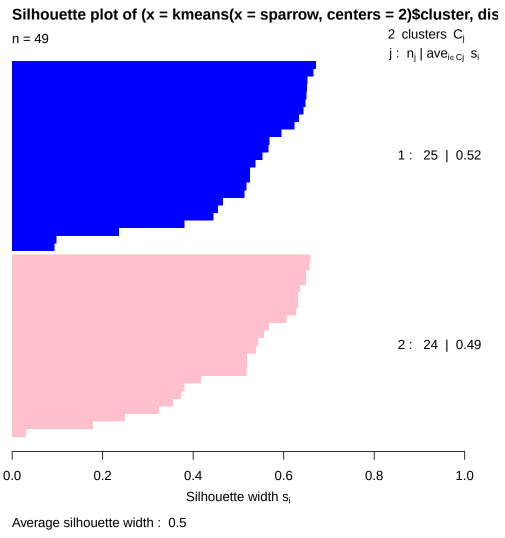
```
## Error in graphics:::plotHclust(n1, merge, height, order(x$order),
hang, :  dendrogramma in input non valido

abline(h = h.max, col = "red")
## with complete linkage the best number of group is two we extract the
## classification with cutree
```

```r
cl <- cutree(tree = h, k = 2) # cut to create two groups cl # first assigned tofirst group
cl <- cutree(tree = h, h = h.max) # cut using an height: the h.max
```

the methods agrees that best method in hierarchical clustering is 2

2.6 K-means

```r
set.seed(1234) ## if we run this function we could get different results, set the seed
km <- kmeans(x = sparrow, centers = 2) # x is the data, we start with two groups

## extract the cluster k-means built
km$cluster # info on clusters: there's not a link between clustering and the

##  [1] 1 2 2 2 2 1 2 2 1 2 2 1 1 1 2 2 1 2 2 1 2 2 2 1 2 1 2 1 1 2 1 1 1 1 2 1 2 1
## [39] 2 1 1 2 2 1 2 1 2 1 1

# actual dead: (dead are the last one)
km$centers # info on centroids

##       totL    AlarE      bhL       hL       kL
## 1 160.9167 245.4583 31.90417 18.80833 21.29583
## 2 155.1600 237.3600 31.03200 18.14400 20.37600

## supposing we dont' know the number of groups: we have to chose it in some
## way

## install.packages("fpc")
library(fpc)

## average silhouette and pearson gamma for several n of k groups
## build a matrix of

indexes <- matrix(NA, nrow = 5, ncol = 2)
for (k in 2:5){
  set.seed(1234)
  km <- kmeans(x = sparrow, centers = k) # centers = k
  stats <- cluster.stats(dist.s, # distance as first object, clustering as second
                        km$cluster)
  indexes[k, ] <- c(
    stats$avg.silwidth, ## average silhouette width in first col
    stats$pearsongamma   ## average pearson gamma second col
  )
}

indexes

##           [,1]      [,2]
## [1,]        NA        NA
## [2,] 0.5046398 0.6496322
## [3,] 0.3375809 0.5476798
## [4,] 0.3925813 0.6049124
## [5,] 0.3952145 0.5493019

## we need to look by column: maximum value for both indicator is k=2 (at least
## with this seed). both indexes agree

## plot silhouette of the two groups
library(cluster)
silhouette

## function (x, ...)
## UseMethod("silhouette")
## <bytecode: 0x55c5646c1590>
## <environment: namespace:cluster>

## silhouette
sil <- silhouette(kmeans(x = sparrow, centers = 2)$cluster,
                  dist.s
                  ) # x is the clustering for the best
plot(sil, col = c("blue", "pink"))
```



```
# silhouette width of eachobservation
```

Capitolo 3

Principal component analysis

With qualitative data correspondance analysis to do the same shit

3.1 Preliminaries

Different from clustering in the sense that from now the focus is on columns of the data matrix (and corresponding summary) with the main ingredient being the linear combinations of the observed variables. For the moment we work at the population level and we see how to apply at the sample level.

A linear combination is sum of vector with weights given from another vector. Assuming we have a random vector X including p random variables (dimension $p \times 1$) a and a vector of constant terms:

$$\underset{p\times 1}{X} = \begin{bmatrix} X_1 \\ \dots \\ X_i \\ \dots \\ X_p \end{bmatrix}, \quad \mathbf{a} = \begin{bmatrix} a_1 \\ \dots \\ a_i \\ \dots \\ a_p \end{bmatrix} \in \mathbb{R}^p$$

a linear combination Y of the two is

$$Y = \mathbf{a}^T X = a_1 X_1 + \ldots + a_i X_i + \ldots a_p X_p$$

We have that Y is a linear combination of the random variables in X with weights given by the components/coefficients of $\mathbf{a}$; Y is a scalar random variable (has dimension 1×1).

We assume that:

- our random vector X has expected value/means $\mathbb{E}[X] = \mu$: each X_i has its own expected values and we collect all these expected value in a vector

$$\mathbb{E}[X] = \boldsymbol{\mu} = \begin{bmatrix} m_1 \\ \dots \\ m_i \dots \\ m_p \end{bmatrix} = \begin{bmatrix} \mathbb{E}[X_1] \\ \dots \mathbb{E}[X_i] \\ \dots \mathbb{E}[X_p] \end{bmatrix}$$

- our vector X has covariance matrix denoted by $\mathbf{\Sigma}$ which is a $p \times p$ matrix with variances on the diagonal and covariances out of the diagonal. Since its is simmetric, square for simplicity people just report the upper matrix.

$$\text{Var}\left[X\right] = \underset{p\times p}{\mathbf{\Sigma}} = \begin{bmatrix} \sigma_1^2 & \dots & \sigma_{12} & \dots & \sigma_{1p} \\ \dots & & & & \\ \dots & & \sigma_2^2 & \dots & \sigma_{2p} \\ \dots & & & & \\ \dots & & & & \sigma_p^2 \end{bmatrix}$$

What happens to expected value and variance/covariance if we take a linear combination of the variable:

- for the expected value we have

  $$\mathbb{E}\left[Y\right] = \mathbb{E}\left[\mathbf{a}^T X\right] = a^T\,\mathbb{E}\left[X\right] = a^T\mu = \mathbf{a_1}\,\mathbb{E}\left[\mathbf{X_1}\right] + \ldots + \mathbf{a_i}\,\mathbb{E}\left[\mathbf{X_i}\right] + \ldots + \mathbf{a_p}\,\mathbb{E}\left[\mathbf{X_p}\right]$$

  so the expected value of a linear combination is the linear combination of the expected values (and is a scalar)

- regardin the variance, to obtain the analog of the square we have to multiply pre and post for $\mathbf{a}$ respecting dimensions

  $$\begin{aligned} \text{Var}\left[Y\right] = \text{Var}\left[a^T X\right] &= \underset{1\times p}{\mathbf{a}^T}\underset{p\times p}{\text{Var}\left[X\right]}\underset{p\times 1}{\mathbf{a}} \\ &= \underbrace{\mathbf{a}^T\mathbf{\Sigma}\mathbf{a}}_{1\times 1} \end{aligned}$$

  so the variance is a scalar as well

This reasoning can be extended to the case where we have more than 1 linear combinations. In that case we can collect the weights in a matrix called $\mathbf{A}$, that will be a pxq where q is the number of linear combinations we're interested in.[1]

A vector of q linear combinations Y $(q \times 1)$ can be obtained as

$$\underset{q\times 1}{Y} = \underset{q\times p}{\mathbf{A}^T}\underset{p\times 1}{\mathbf{X}}$$

Properties of mean and variance of transformations applies again so

$$\begin{aligned} \mathbb{E}\left[Y\right] &= \mathbf{A}^T\,\mathbb{E}\left[X\right] = \mathbf{A}^T\boldsymbol{\mu} \\ \text{Var}\left[Y\right] &= \mathbf{A}^T\,\text{Var}\left[X\right]\mathbf{A} = \mathbf{A}^T\mathbf{\Sigma}\mathbf{A} \end{aligned}$$

with $\mathbb{E}\left[Y\right]$ a $q \times 1$ vector and $V(Y)$ no longer a scalar but a $q \times q$ matrix (Since we have more than one random variable there is not only the variances but also the covariances)

[1]Warning: we've no enough letters so the notation is consistent within each topic. This matrix $\mathbf{A}$ has to do nothing with the centering matrix seen before. In this chapter $\mathbf{A}$ is the matrix of coefficients

Example 3.1.1. Assume we have a bivariate random vector X, $X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$ with mean $\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$ and variance/covaraince matrix $\mathbf{\Sigma} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{bmatrix}$.
Now let's consider the following linear combinations

$$\begin{aligned} Y_1 &= X_1 - X_2 \\ Y_2 &= X_1 + X_2 \end{aligned}$$

Derive the expected values and the covariance matrix of $Y = \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}$

Example 3.1.2. Consider three independent standardized variables Z_1, Z_2, Z_3. Assume we transform them as

$$\begin{aligned} Y_1 &= z_1 \\ Y_2 &= Y_1 + 0.01 Z_2 \\ Y_3 &= 10 Z_3 \end{aligned}$$

derive the covariance matrix of $Y = \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \end{bmatrix}$

Back on the variance of a single linear combination we've said that

$$\text{Var}\left[Y\right] = \text{Var}\left[\mathbf{a}^T X\right] = \mathbf{a}^T \mathbf{\Sigma} \mathbf{a}$$

considering the simple case of X composed by two variable $X = [X_1, X_2]^T$, this means that

$$\begin{aligned} \text{Var}\left[Y\right] = \mathbf{a}^T Sigma\mathbf{a} &= \underbrace{\begin{bmatrix} a_1 & a_2 \end{bmatrix}}_{1\times 2} \underbrace{\begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{bmatrix}}_{2\times 2} \underbrace{\begin{bmatrix} a_1 \\ a_2 \end{bmatrix}}_{2\times 1} \\ &= \begin{bmatrix} a_1\sigma_1^2 + a_2\sigma_{12} & a_1\sigma_{12} + a_2\sigma_2^2 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \\ &= a_1^2 + \sigma_1^2 + a_1 a_2 \sigma_{12} + a_1 a_2 \sigma_{12} + a_2^2 \sigma_2^2 \\ &= a_1^2 \sigma_1^2 + 2 a_1 a_2 \sigma_{12} + a_2^2 \sigma_2^2 \end{aligned}$$

Focusing

- on the first element a_1 (taken as unknown) our interest is to know what happens when a_1 changes (we still don't know what these coefficients are). Thinking variance as function of a_1 what happens to the variance when a_1 changes? This is an upward parabola (coefficient σ_1^2 is necessarily positive being a variance) : as a_1 varies the variance describe an upward parabola
- focusing on a_2 it's the same thing, again it's an upward parabola

- so in our bidimensional space a_1, a_2 as x, y we have a cup with relative to variance.
  This variance goes to infinity, does not have a finite maximum: we can increase the variance of linear combination just by simply increasing the value of a_1 and a_2 or, in other words, in the multidimensional space the variance of a linear combination can be increased just by using a vector **a** with norm higher and higher.
  the larger the norm of the vector $\mathbf{a} = (a_1, a_2)$ the larger the variance. So the variance of a linear combination does not have a finite maximum. Keeping the data as fixed we can increase the variance of a linear combination by simply increasing the norm of the vector a containing coefficient of the linear combination **a**
  This is something we don't like: we want something that is linked to the data for an increase in variance. This is a point important for principal component.

TODO: fare grafici

Important remark 8. To take the linear algebra approach the variance $\mathrm{Var}[y] = \mathbf{a}^T \Sigma \mathbf{a}$ is a quadratic form, represented by a positive semidefinite matrix.
So a positive semidefinite quadratic form does not have a finite maximum.

Remark 9. These are Two different way of saying the same and are the preliminaries for PCA

3.2 PCA

Principal component analysis has been invented by Pearson in 1901, but the way we'll see it is not based on the developmente of Pearson, but using the approach of Hotelling in 1933. They wanted to solve two different problem but found to obtain the same solutions, they've both invented it:

- Pearson wanted to solve a regression problem (1901 "On lines and planes of closest fit to a system of points"); he wanted to find a line that best fit the point, but in a special condition where he had errors in the predictors (while in regression models one assumes that the x's are fixed without errors, Pearson was in a different situation having error in x). It was something completely different from motivating PCA methods but obtained the same

TODO: CHECK 05 18/2

- Hotelling OTOH have too many variables and can't find a way to manage them: how can I summarize the variables producing a smaller number of variables that keep as much info as in the original sets? Hotelling find a low dimensional function (linear combination) of the observed data that preserves as much of the variability as possible The approach is the same used at today. Method is also known as dimension reduction

They didn't have machines to do calculations, but used cleverly linear algebra, eigenvalue and eigenvector to tackle the problem.

Let's start by considering a single linear combination of the variables (we see other combination afterward); we have

$$y_1 = \mathbf{a}_1^T X$$

where $\mathbf{a}_1$ is a vector, not an element.
The point is that we want to find a linear combination (so determines the values of a_1), such that the variance of the combination is maximum

$$V(y_1) = \mathbf{a}_1^T Sigma \mathbf{a}_1 \quad \text{is maximized}$$

Problem is that, as we've seen before, a maximum on variance does not exists: it is enough to take $\mathbf{a}_1$ longer and longer.
What interests us is actually the direction of the vector $\mathbf{a}_1$: in that case I can put a contraint on the norm of $\mathbf{a}_1$ (to have a fair comparison on variances that depends only on direction we put a constraint).
In this case a unit norm vector is enough to identify a direction in space, so the constraint that we put is that $\|a_1\| = 1$, that is $\mathbf{a}_1^T\mathbf{a}_1 = 1$.
So finally he problem we need to solve is a *constraint optimization problem*. That is we want to choose $\mathbf{a}_1$ to maximize $V(Y) = \mathbf{a}_1^T\Sigma\mathbf{a}_1$ under the constraint $\|a\| = a_1^T a_1 = 1$.
This can be solved by using lagrange multiplier: the problem above is equivalent to maximize this function below

$$\phi = \mathbf{a}_1 \Sigma a_1 - \lambda_1(\mathbf{a}_1^T\mathbf{a}_1 - 1)$$

where λ_1 is a Lagrange multiplier.

TODO: check 05

Now we need to maximize by taking derivatives $\mathbf{a}_1$ is our unknown; we check/consider all possible unit norm vector, and put a system of p equations (we need to take p different derivatives, one for each element of $\mathbf{a}_1$) where

$$\frac{\partial\phi}{\partial\mathbf{a}_1} = 2\Sigma\mathbf{a}_1 - 2\lambda_1\mathbf{a}_1$$

which equaled to $\mathbf{0}$ yields

$$\Sigma a_1 = \lambda_1 a_1$$

That is we need to find the $\mathbf{a}_1$ that satisfy the equation above so the solution to the problem consist in a pair of eigenvalue-eigenvector for the matrix Sigma.
This above is identity relationship between eigenvalues and eigenvectors: this holds if $\mathbf{a}_1$ is eigenvector of Σ and λ_1 is the corresponding eigenvalue.
How many eigenvalues has Σ? p, since it'is $p \times p$ so it is expected to have p. Point is that we need just 1: which pair of eigenvalues-vector solves my problem?
In order to answer let's premultiply both sides by $\mathbf{a}_1^T$

TODO: revisit spectral decomposition

$$\begin{aligned}
\Sigma\mathbf{a}_1 &= \lambda_1\mathbf{a}_1 \\
\mathbf{a}_1^T\Sigma\mathbf{a}_1 &= \lambda_1 \underbrace{\mathbf{a}_1^T\mathbf{a}_1}_{=1} \\
\mathbf{a}_1^T\Sigma\mathbf{a}_1 &= \lambda_1
\end{aligned}$$

this is the answer: to maximize the variance, which is the right quantity on the left we need to take the largest eigenvalue. The eigenvalue coincides with the variance af the linear transformation.
in order to have *maximum variance* we need to consider the *largest eigenvalue* of Σ and $\mathbf{a}_1$ will be the corresponding eigenvector.

The linear combination of having $\mathbf{a}_1$ as vector of coefficients is called *first principal component*:

$$\mathbf{y}_1 = \mathbf{a}_1\mathbf{X}$$

In other words we can say that the first principal component is the linear combination of the observed variables *having the largest variance*. It's variance will be the largest eigenvalue of Σ and the optimal vector of coefficients $\mathbf{a}_1$ will be the corresponding eigenvector.

TODO: CHECK 05

Main point so far:

- Principal Components are linear combinations of observed variable: main tool we use is linear combinations.
- Any linear combination does not have finite variance; the variance is positive semidefinite (Sigma) and has no finite maximum. Variance is an upward parabola.
- Maximizing without constraint is therefore risky, we could increase fictitiously variance without reference to the data explained. What we're interested in is not the norm of the vector but the direction of the vector; the unit norm is enough to identify direction in space and to make fair comparison we compare vector having the same norm, for simplicity we take unit norm.
- We want to maximize the variance under the constraint looking only at unit norm vectors of combination. We use Lagrange multiplier to do the maximization.
- To find the maximum we take first derivative equal to 0: our function has a maximum because we constraint on a unit norm vector the maximum is reached with eigenvalue/eigenvector equality
- Problem is to find suitable pair of eigenvalue eigenvector: which one to consider? the higher, since it is the variance So we've identified the first principal component

We've spoken of the first principal components: maybe it's very improbable that a single linear combination is enough and we need more than one.

We've said that Σ has p pairs eigenvalue-eigenvector: with large number of variables it is difficult that a single combination is enough to save all the variability of the higher dimensional space. So we're interested in looking at a second linear combination.
How can we find the second principal component?
We need to constraint the optimization search in the orthogonal space to what found at the first passage (we want the PC to be orthogonal/uncorrelated).
Our second component will be

$$Y_2 = \mathbf{a}_2^T\mathbf{X}$$

with variance

$$\text{Var}\left[y_2\right] = \mathbf{a}_2^T\Sigma\mathbf{a}_2$$

our lagranbge function will be

$$\phi = \mathbf{a}_2^T \Sigma \mathbf{a}_2 - \lambda_2(\mathbf{a}_2^t \mathbf{a}_2 - 1) - \lambda_3(a_1^t a_2)$$

above:

- the first constraint to have unit norm in $\mathbf{a}_2$
- the second is to impose that $\mathbf{a}_1^T \mathbf{a}_2 = 0$ (we removed -0) so that $\mathbf{a}_1$ and $\mathbf{a}_2$ are orthogonal

This amounts to maximize $\mathbf{a}_2^T \Sigma \mathbf{a}_2$ under the unit norm constraint $\mathbf{a}_2^T \mathbf{a}_2 = 1$ and under the constraint that $\mathbf{a}_2$ is orthogonal to $\mathbf{a}_1$, that is $\mathbf{a}_1^T \mathbf{a}_2 = \mathbf{a}_2^T \mathbf{a}_1 = 0$. In order to find $\mathbf{a}_2$ we can derive ϕ respect to $\mathbf{a}_2$ which is our unknown

$$\frac{\partial \phi}{\partial \mathbf{a}_2} = 2\Sigma a_2 - 2\lambda_2 \mathbf{a}_2 - \lambda_3 \mathbf{a}_1$$

then we set our derivative equal to $= 0$. This problem can be simplified a little bit. Let's premultiply both sides by $\mathbf{a}_1^T$, we have that

$$2\mathbf{a}_1^T \Sigma \mathbf{a}_2 - 2\lambda_2 \mathbf{a}_1^T \mathbf{a}_2 - \lambda_3 \underbrace{\mathbf{a}_1^T \mathbf{a}_1}_{=1} = 0$$

Now considering $\mathbf{a}_1^T \Sigma$ we have

$$\mathbf{a}_1^T \Sigma = (\Sigma \mathbf{a}_1)^T = \lambda_1 \mathbf{a}_1^T$$

so it is related to the first PC. Back to our main equation

$$2\lambda_1 \underbrace{\mathbf{a}_1^T \mathbf{a}_2}_{=0} - 2\lambda_2 \underbrace{\mathbf{a}_1^T \mathbf{a}_2}_{=0} - \lambda_2 \underbrace{\mathbf{a}_1^T \mathbf{a}_1}_{=1} = 0$$

$$-\lambda_3 = 0$$

This implies that $\lambda_3 = 0$ Back to our derivative set equal 0 this means that therefore the derivative of ϕ wrt to $\mathbf{a}_2$ will be

$$\frac{\partial \phi}{\partial \mathbf{a}_2} = 2\Sigma \mathbf{a}_2 - 2\lambda_2 \mathbf{a}_2 = 0$$

$$\Sigma \mathbf{a}_2 = \lambda_2 \mathbf{a}_2$$

and this is again an eigenvalue-eigenvector relationship. If we premultiply by $\mathbf{a}_2^T$

$$\underbrace{\mathbf{a}_2^T \Sigma \mathbf{a}_2}_{\mathrm{Var}[Y_2]} = \lambda \underbrace{a_2^T a_2}_{=1}$$

so

$$\mathrm{Var}\,[Y_2] = \lambda_2$$

as we want the linear combination of x having the largest variance in the orthogonal complement of $\mathbf{a}_1$ we take λ_2 as the second largest eigenvalue and $\mathbf{a}_2$ will be the corresponding eigenvector.
The second PC is the linear combination of X with $\mathbf{a}_2$ (the eigenvector corresponding to second largest eigenvalue).
Goin on like this, for p variables we can find up to p principal components.

3.3 Properties

Having used an orthogonal $\mathbf{a}_2$ we obtain something interesting, PC s have a very relevant stratical property. Let's consider our

$$\Sigma \mathbf{a}_2 = \lambda_2 \mathbf{a}_2$$

premultiplying by $\mathbf{a}_1^T$

$$\mathbf{a}_1^T \Sigma \mathbf{a}_2 = \lambda_2 \mathbf{a}_1^T \mathbf{a}_2$$

but we know that for orthogonality constraint $\mathbf{a}_1^T \mathbf{a}_2 = 0$ so

$$a_1^T Sigmaa_2 = 0$$

what is this? This is the covariance between Y_1 and Y_2, the covaraince between the first two principal components. Let's prove it. Having

$$y_1 = \mathbf{a}_1^T X$$
$$y_2 = \mathbf{a}_2^T X$$

without loss of generality let's assume for simplicity that X variables are mean centered. This implies that the covariance between Y_1 and Y_2 is equal to

$$\text{Cov}(Y_1, Y_2) = \mathbb{E}[Y_1 \cdot Y_2] - 0$$

but

$$\mathbb{E}[Y_1 \cdot Y_2] = \mathbb{E}\left[\mathbf{a}_1^T X X^T \mathbf{a}_2\right] = a_1^T \, \mathbb{E}\left[X X^T\right] a_2$$

where $XX^T = \Sigma$ for what seen in statistics.
This means that principal compoenents are uncorreleted (covariance is numerator of correlation). Thus the first and second principal component are uncorrelated (this is a not scontated gift, under the constraint imposed).
The fact that orthogonal vectors generate uncorrelated variable is not necessarily true.
As an exercise think a counterexample: an example in which you have linear combination of the observed variable that are identified by orthogonal vectors and are correlated
Orthogonality and uncorrelation are not sinonyms

Example 3.3.1. A counterexample. Consider the vectors of the canonic bases

$$\mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \mathbf{e}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Any point in $\mathbb{R}^2$ can be obtained as combination of those two.
we're interested in a generic (x_1, x_2) we can think of coordinates as linear combination of vector from canonic base

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = x_1 \mathbf{e}_1 + x_2 \mathbf{e}_2 = \begin{bmatrix} x_1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

OO: grafico

each point on a cloud of correlated variables x_1 and x_2 can be generated by $\mathbf{e}_1, \mathbf{e}_2$; $\mathbf{e}_1$ and $\mathbf{e}_2$ are orhogonal but the corresponding linear combination are correlated
The points can be thought of as linear combination with coefficients given by the vectors of the canonical basis (which are orthogonal by definition) but the variables are correlated.
On the contrary n PC orthogonal vectors (eigen) and UNCORRELATED variables

3.4 Obtaining PCs in practice

In practice how to obtain principal components?

$$\Sigma \mathbf{a}_1 = \lambda_1 \mathbf{a}_1$$
$$\Sigma \mathbf{a}_1 - lambda_1 \mathbf{a}_1 = \mathbf{0}$$

now we collect $\mathbf{a}_1$ obtaining the identity matrix of the same size

$$(\Sigma - \lambda_1 \mathbf{I})\mathbf{a}_1 = \mathbf{0}$$

our problem is to find $\mathbf{a}_1$ which solves this equation.
At left we have a matrix (within parenthesis) times unknown which is equal to 0. It's a linear equation system which is homogeneous (constant term is 0). When does a linear equation system admit a nontrivial solutions (eg $\mathbf{a}_1 = \mathbf{0}$)? how should the matrix of the coefficient be?

In order for an homogenous equation system to have a nontrivial solutions (ie $\mathbf{a}_1 \neq 0$) the matrix of coefficient needs to be singular: that is its determinant needs to be equal to zero. This means that λ_1 should be *a root of the characteristic polinomial*.

So finding the principal components amounts to solve eigenvalue/vector problem or to solve a linear eq system.

$(\Sigma - \lambda_1 \mathbf{I})$ has to be singular, so λ_1 is a root of the characteristic polynomial. **TODO**: CHECK 19/2

Example 3.4.1. Considering a bivariate vector

$$X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}, \quad \operatorname{Cov}(X) = \Sigma = \begin{bmatrix} 5 & 2 \\ 2 & 2 \end{bmatrix}$$

we want to find the principal components. What we need to solve is

$$(\Sigma - \lambda_1 \mathbf{I})\mathbf{a}_1 = \mathbf{0}$$

what I do is to write

$$\left[\begin{bmatrix} 5 & 2 \\ 2 & 2 \end{bmatrix} - \lambda_1 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right] \mathbf{a}_1 = \mathbf{0}$$

where λ_1 is our unknown and $\mathbf{a}_1$ the vector of coefficients we're looking for. We rewrite matrix coefs as

$$\begin{bmatrix} 5 - \lambda_1 & 2 \\ 2 & 2 - \lambda_1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

where for notation simplicity $\mathbf{a}_1 = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$.
In order to have a nontrivial solution we need that determinant to be null

$$\begin{aligned}
&\begin{vmatrix} 5-\lambda_1 & 2 \\ 2 & 2-\lambda_1 \end{vmatrix} = 0 \\
&(5-\lambda_1)(2-\lambda_1) - 2 \cdot 2 = 0 \\
&10 - 5\lambda_1 - 2\lambda_1 + \lambda_1^2 - 4 = 0 \\
&\lambda_1^2 - 7\lambda_1 + 6 = 0 \\
&(\lambda_1 - 6)(\lambda_1 - 1) = 0 \\
&\lambda_1 = 6, \lambda_2 = 1
\end{aligned}$$

these are the two eigenvalue of Σ, 6 and 1, which are the variances of the two principal components. 6 will be the variance of the first PC and 1 the variance of the second one.
Let's appreciate two relevant aspects:

1. back to our original covariance matrix Σ: 6 is larger than any entries on the diagonal (5 and 2). That is: the variance of the first principal component (6) is higher than any variance of the original variable.
   In general the variance of the first PC will *never* be smaller than the largest observed varaince: it can be that the variance is equal but will be never be below

2. looking at the trace of Sigma: 5+2. If we go back to our variance the sum is still 6+1=7. This always holds, so

$$\sum_{i=1}^{p} \text{Var}[X_i] = \sum_{k=1}^{p} \lambda_k$$

Remembering that the PCs are uncorrelated, we can write the variance covariance matrix of the PC as a matrix Λ:

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

the off diagonal elements is always = 0 (being PCs uncorrelated). Thus we can rewrite

$$\begin{aligned}
&\sum_{i=1}^{p} \text{Var}[X_i] = \sum_{k=1}^{p} \lambda_k \\
&\text{Tr}\,\Sigma = \text{Tr}\,\Lambda
\end{aligned}$$

So if one sum the eigenvalue variance of principal components it get the same variaiblity of the orginal variable (trace of varcov).
Through PC we obtain a different way to split variability, we are not changing the total variability.

TODO: CHECK (19/2)

Back to solve our problem we want to find principal components: once found our 6 an 1 eigenvalue we want to use them to obtain eigenvector. Back to our linear system:

$$\begin{bmatrix} 5-\lambda_1 & 2 \\ 2 & 2-\lambda_1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Now since $\lambda_1 = 6$ (consider first PC) we have

$$\begin{bmatrix} 1 & 2 \\ 2 & -4 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{cases} -a_1 + 2a_2 = 0 \\ 2a_1 - 4a_2 = 0 \end{cases}$$

In order now to solve this system of equations (eg say by substitution) we run in first problem: the second equation is the first one multiplied by -2: so we have a single equation with 2 unknown.
But up to now we didn't used the unit norm constraint, which become useful/comes into play, so we add to the first equation

$$-a_1 + 2a_2 = 0$$
$$a_1^2 + a_2^2 = 1 \quad (\mathbf{a}_1^T\mathbf{a}_1 = 1)$$

Now we use substitution taking a_1 in the first equation

$$\begin{cases} a_1 = 2a_2 \\ 4a_2^2 + a_2^2 = 1 \end{cases} \quad \begin{cases} ,, ,, \\ 5a_2^2 = 1 \end{cases} \quad \begin{cases} ,, ,, \\ a_2^2 = \frac{1}{5} \end{cases} \quad \begin{cases} ,, ,, \\ a_2^2 = \pm\sqrt{\frac{1}{5}} \end{cases}$$

I decide to take the positive root so

$$\begin{cases} a_1 = \frac{2}{\sqrt{5}} \\ a_2 = \frac{1}{\sqrt{5}} \end{cases} \implies \mathbf{a}_1 = \begin{bmatrix} \frac{2}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} \end{bmatrix}$$

This is my vector $\mathbf{a}_1$ with component $2/\sqrt{5}$ and $1/\sqrt{5}$. this is the vector of coeffiicvents of the first principal components!

One can either/also take the negative root/solution obtaining

$$\begin{cases} a_1 = -\frac{2}{\sqrt{5}} \\ a_2 = -\frac{1}{\sqrt{5}} \end{cases}$$

These vector are the same for our interest, the direction (pendenza) is the same: it depends on the software we use which solution is provided   Thus principal component are uniquely defined up to sign changes (which can change from software to software.

TODO: CHECK 19/2

Let's calculate the second PC. Considering $\lambda_2 = 1$ we have

$$\begin{bmatrix} 5-1 & 2 \\ 2 & 2-1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{cases} 4a_1 + 2a_2 = 0 \\ 2a_1 + a_2 = 0 \end{cases}$$

Again we take one of the two equations only (the second) since are linear combinations and add the unity norm constraint

$$\begin{cases} 2a_1 + a_2 = 0 \\ a_1^2 + a_2^2 = 1 \end{cases} \quad \begin{cases} a_2 = -2a_1 \\ a_1^2 + 4a_1^2 = 1 \end{cases} \quad \begin{cases} ,,,, \\ 5a_1^2 = 1 \end{cases} \quad \begin{cases} ,,,, \\ a_1^2 = \pm\frac{1}{\sqrt{5}} \end{cases}$$

Taking the positive root we end with

$$\begin{cases} a_1 = \frac{1}{\sqrt{5}} \\ a_2 = -\frac{2}{\sqrt{5}} \end{cases} \implies \mathbf{a}_2 = \begin{bmatrix} \frac{1}{\sqrt{5}} \\ -\frac{2}{\sqrt{5}} \end{bmatrix}$$

One thing to appreciate: while the vector of first component have all positive elements (or all negative) if we want an ortogonal second component we need something that null the product of the two vectors so some elements will be positive and some negative.

In general if first PC is a strict linear combinations (all entries with same sign), then the following (second and 3rd and so on) need/will be a contrast (different sign) in order orthogonality to be satisfied (vector product will be 0).

Finally we can collect all eigenvector in a matrix called $\mathbf{A}$, whose columns are the eigenvector of Σ in decreasing order of corresponding egenvalue

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \end{bmatrix} = \begin{bmatrix} \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{5}} \\ \frac{1}{\sqrt{5}} & -\frac{2}{\sqrt{5}} \end{bmatrix}$$

Thus we can write in matrix form the vector of principal components

$$\mathbf{Y} = \mathbf{A}^T\mathbf{X}$$

where now Y is the vector of the principal component which is $p \times 1$

In general $\mathbf{A}$ is a $p \times p$ matrix whose columns are the eigenvector of Σ ordered according to decreasing values of the corresponding eigenvalues.

$$\underset{p\times p}{A} = \begin{bmatrix} \mathbf{a}_1 & a_2 & \dots & \mathbf{a}_p \end{bmatrix}$$

Thus matrix $\mathbf{A}$ has orthogonal columns and actually

$$\mathbf{A}^T\mathbf{A} = \mathbf{I}$$

this because eigenvector have unit norm (so in the diagonal $\mathbf{a}_i^T\mathbf{a}_i = 1$) and are orthogonal (so out of main diagona $\mathbf{a}_i^T\mathbf{a}_j = 0$).
Based on the matrix $\mathbf{A}$ we can define the vector of principal components Y which is $p \times 1$

$$\underset{p\times 1}{Y} =$$

$$mathbf A\mathbf{X}$$

Y is the vector of the principal components, while X is the vector of the observed variables.
In one of our first calsses we seen that

$$\mathrm{Var}\left[Y\right] = \mathbf{A}^T\,\mathrm{Var}\left[X\right]\mathbf{A}$$

and we know that $\mathrm{Var}\left[(\right] X) = \Sigma$ so

$$\mathrm{Var}\left[Y\right] = \underbrace{\mathbf{A}^T}_{p\times p}\underbrace{\Sigma}_{p\times p}\underbrace{\mathbf{A}}_{p\times p}$$

covariance of pairs of principal component is also 0

$$\underset{p\times p}{\Lambda} = \begin{bmatrix} lambda_1 & \dots & 0 & \dots & 0 \\ \dots & & & & \\ 0 & \dots & \lambda_i & \dots & 0 \\ \dots & & & & \\ 0 & \dots & 0 & \dots & \lambda_p \end{bmatrix}$$

Lambda has eigenvalue in decreasing order and out of diagonal we have 0

$$\left[\mathrm{Var}\left[Y\right] = \mathbf{A}^T\Sigma\mathbf{A} = \Lambda\right]$$

we take a Σ full matrix and diagonalized it (transformed in two diagonal matrix: that is spectral decomposiztion/theorem)

TODO: CHECK 07

Important remark 9 (Similar matrices (reminder)). Two matrices $\mathbf{X}$ and $\mathbf{Y}$ are similar if there exists a non-singualar matric $\mathbf{Z}$ such that $\mathbf{Z}^{-1}\mathbf{Y}\mathbf{Z} = \mathbf{X}$.

In our case we have that

$$A^T SigmaA = Lambda$$

so if Λ takes the role of X and Σ the role of Y, in definition above we have that Σ and Λ **are similar** (since $\mathbf{A}^T = \mathbf{A}^{-1}$ being $\mathbf{A}$ orthogonal)

Similar matrices share many properies: among them they have the same trace. So $\mathrm{Tr}\, Sigma = \mathrm{Tr}\, Lambda$ and this is the proof that variances of observed variances coincides with sum of variances of principal components.

Thus PC transform data but doesnt change total variability.

Actually what PCA does is to apply the spectral decomposition to covariance matrix: this is interesting from the statistical pov in order to have uncorrelated new variables and same total variances (represented by eigenvectors).

3.5 Interpretation

What does we do when we perform PC. (Let's assume variables are mean centered for simplicity)

TODO: grafico

Assume we want to rotate the reference system. I take the axis and then an orthogonal (in red) which are Y_1 and Y_2.
The points will be transported in the new reference system and will be transformed to y_1 and y_2. clearly the coordinates are linked.

The coordinates on the new sistem depend on the angle between red and blue line (α). if one change alpha, change the coordinates of the points if we remember high school the relationship between coordinates of two reference systems are

$$\begin{cases} y_1 = \tilde{x}_1 \cos\alpha + \tilde{x}_2 \sin\alpha \\ y_2 = -\tilde{x}_1 \sin\alpha + \tilde{x}_2 \cos\alpha \end{cases}$$

but $\cos^2 \alpha + \sin^2 \alpha = 1$

so every component is unit norm vector.

We can collect the coefficient

$$\mathbf{a}_1 = \begin{bmatrix} \cos\alpha \\ \sin\alpha \end{bmatrix}, \quad \mathbf{a}_2 = \begin{bmatrix} -\sin\alpha \\ \cos\alpha \end{bmatrix}$$

thus those two vector have unit norm. moreover

$$\mathbf{a}_1^T \mathbf{a}_2 = -\cos\alpha\sin\alpha + \sin\alpha\cos\alpha = 0$$

This means that we can think PCA as an **orthogonal rotation of the reference system**.

We have an infinity of possible orthogonal rotations (rotations are infinite): the one corresponding to PC is such that the **spread on the point of the first component is the maximum**.
Performing PC is just changin the reference system: actually we choose the optimal α with PC; trace remains the same because we're just rotating the system not changing data.

It's just a data transformation (orthogonal rotation) procedure and always is feasible: one can always move from X to Y. (OTOH factor analysis has many things in common but it is a model and not necessarily exists or appropriate)

TODOHERE

Reason of the popularity is because it allows us to collapse multidimensionality; ege in a situation like this (mean centered varaible for simplicity)

TODO: Immagine

Along the first pc the variability is higher, along the second a bit less: the core of data is along Y_1, remaining part could be noise.

In general however we need formal criteria to choose the *number of components* which are enough to summarize the data. There are different answers to this. Before considering this let's see something more on PC which is a sad side of the story

Dependance on units of measure

Consider the following situations with two covariance matrix equal in all but in the units

$$\Sigma_1 = \begin{bmatrix} 90 & 50 \\ 50 & 90 \end{bmatrix}, \quad \Sigma_2 = \begin{bmatrix} 9000 & 500 \\ 500 & 90 \end{bmatrix}$$

The point is: data are the same but in the first case X_1 is measured in cm, in the second case it is expressed in mm. Does the change in the measurement unit affect PCA results?
If we do spectral decomposition the following is obtained; for Σ_1

$$y_1 = 0.707x_1 + 0.707x_2$$

and $\lambda_1 = 140$. As we can see

- the variables are equally weighted: the two variablesa are equally important in the first component

- trace of covariance matrix is 180, while variance explained by first PC is 140. Considering the first PC only, we take into account 78 of total variability.

For Σ_2

$$y_1 = 0.998x_1 + 0.055x_2$$

with $\lambda_1 = 9027.97$ and percentage explained is $\lambda_1 / \operatorname{Tr} \Sigma_2 = 99.32$ %. So in this case:

- coefficients are very different: the combination is completely dominated by x_1, while x_2 has a small coeff
- the explained variance increase just because of the unit measure and the increased importance of x_1 in the PC solution

This example tells us that *PCA is not scale equivariant*: it depends on units of measurement. The solutions we obtain can be completely different.

Often we don't want this to happen, our result to be dependent of the unit of measure.

So standard practice is to standardize before (and considering the correlation matrix) if we don't want results to be affected by units of measure. **TODO**: CHECK 07

For PCA this amounts to obtain eigenvalues and eigenvectors (see first lessons) from the correlation matrix ρ (instead of the covariance), as we have seen that the covariance matrix of the standardized variables is the correlation matrix.

Is there a relationship between eigenvalues/vectors of Sigma and Corr? In this case I could move from one solutions to the other? We know that Rho

$$Rho = \Delta^{-1/2}\Sigma\Delta^{-1/2}$$

where Δ is analog of D and Σ is analog of S? and

$$\Delta = \begin{bmatrix} \sigma_1^2 & \dots & 0 & \dots & 0 \\ \dots \\ 0 & \dots & \sigma_i^2 & \dots & 0 \\ \dots \\ 0 & \dots & 0 & \dots & \sigma_p^2 \end{bmatrix}$$

is obtained on the sample data.

Now if a is an eigenvector of ρ and λ is the corresponding eigenvalue, we have for what seen so far

$$\rho\mathbf{a} = \lambda\mathbf{a}$$

(relationship be eigenval/vecs) If b is an eigevector of $Sigma$ and δ is the corresponding eigenvalue then we have that

$$\Sigma\mathbf{b} = \delta\mathbf{b}$$

(same eigenvalue/vector relationship) but from

$$\rho = \boldsymbol{\Delta}^{-1/2}\boldsymbol{\Sigma}\boldsymbol{\Delta}^{-1/2} \implies \boldsymbol{\Sigma} = \boldsymbol{\Delta}^{1/2}\rho\boldsymbol{\Delta}^{1/2}$$

So we have a new way to write the covariance matrix which show the **link between covariance and correlation matrix**

So

$$\Sigma \mathbf{b} = \delta \mathbf{b}$$
$$Delta^{1/2} Rho Delta^{1/2} b = deltab$$

if we premultiply both sides to $\Delta^{-1/2}$ on the left hand side we obtain

$$Rho Delta^{1/2} \mathbf{b} = delta Delta^{-1/2} \mathbf{b}$$

now we decide that $\mathbf{c} = \Delta^{1/2} \mathbf{b}$ so

$$\rho \mathbf{c} = \delta \boldsymbol{\Delta}^{-1} \mathbf{c}$$

we end with $\mathbf{c}$ which is eigengevcor of ρ only if $\Delta^{-1} \mathbf{c}$ is $= \mathbf{c}$.
But $\Delta^{-1} \mathbf{c} neq \mathbf{c}$ because Δ is not the identity matrix (its the matrix of variances of variables). So

$$\rho \mathbf{c} = \delta \boldsymbol{\Delta}^{-1} \mathbf{c}$$

is not an eigenvalue-eigenvector relationship. This tells us that the eigenvalues and eigenvectors of Sigma are different from the eigenvalues/vectors of Rho and there is no way to move from one solutions to the other.

Important remark 10. So NO, there's no relationship between eigenvalues/vectors of Σ and ρ. When performing PCA we need to decide *before* to standardize or not: if we have

1. different units of measuremente or
2. very different variances

then standardize variables is better.
Otherwise if differences in variances are important and we don't want to homogeneize it, then use the covariance unstandardized matrix.

Example 3.5.1. Consider

$$\rho = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$$

derive principal components, what changes when ρ is positive or negative?

3.6 Number of components to choose

TODO: SAMPLE LEVEL (RECHECK) 08

Now we look on how to decide how many components we need/take to at the sample level. When we're given sample data we need to estimate Σ or Rho with ML estimates:

- for Σ we use S (the ML estimates for Σ, dividing codeviances by n). If we prefer we could use the unbiased estimate for Σ (dividing codeviances by $n-1$ instead of n); the variance of the principal components will be rescaled accordingly and the eigenvectors will not change;

- for *Rho* we use R where the same happens.

DO: CHECK 08 25/2

How can we decide how many PCs we should keep in order to reduce dimensionality and at the same time preserve as much info as possible? We have a tradeoff: we want to reduce dimensionality but don't want to loose to much information.

We have different criteria to select n of principale components, all criteria are rules of thumb to select n of PC (not formal criteria, suggestion based on empirical evidence).

Three different rule of thumb exists:

1. percentage of explained/retained variance
2.
3.

The reason why there's no formal criteria is because so far we made no assumption on distribution of variables: PC are data transformation and such can be always obtained, it's just an algebraic procedure stuff done on covariance matrix or correlation matrix. In principle we don't need info on population: as it's very general the only way we have to make decision is to use empirical rules (in essence it's a descriptive method).

The criteria depends on if we work with S or R (meaning is the same but from practical pov we do different things) so we need to decide in advance if we want to use S or R:

3.6.1 Percentage of explained variance

Common to both strategies (starting from S or R) is cumulative percentage of explained/retained variance.

For what concerns:

- S: we know that the

  $$\operatorname{Tr} S = \operatorname{Tr} L = \sum_{i=1}^{p} s_i^2 = sum_{k=1}^{p} l_k$$

  where L is the matrix with the eigenvalues (L is the analog of Λ in the sample) s_i^2 is an observed variance.
  The proportion of total variance explained by first PC is

  $$\frac{l_1}{\operatorname{Tr} L} \cdot 100$$

  The proportion explained by first two PCs is

  $$\frac{l_1 + l_2}{\operatorname{Tr} L} \cdot 100$$

  The percentage explained up to m components is

  $$\frac{\sum_{k=1}^{m} l_k}{\operatorname{Tr} L} \cdot 100$$

  The criteria is to stop cumulating at the value m that make the above explained $\geq 80\%$

- R: similarly we have that

$$\operatorname{Tr} R = \operatorname{Tr} L = \sum_{i=1}^{p} 1 = sum_{k=1}^{p} l_k$$

where we substituted the variance s_i^2 with 1, that is the correlation of each variable with itself. So with standardized data the total variance ($\operatorname{Tr} R$)is just the number of variables p.
The reasoning is the same but percentage of explained by first component is now

$$\frac{l_1}{p} \cdot 100$$

And again the final criteria is

$$\frac{\sum_{k=1}^{m} l_k}{p} \cdot 100 \geq 80\%$$

TODO: riascolta: has two che cosa?

R has two (one based on spectral decomposition one based on Singular value decomposition but eigenvector are the same)

3.6.2 Threshold on eigenvalues (Kaiser's rule)

For the second rule, known as Kaiser's rule, we start seeing this applied correlation matrix (it was invented when working with standardized variable).
For:

- R: we keep as many principal components as are the eigenvalues ≥ 1: in order to keep the principal components that do better than original variabiles (which have variance 1 being standardized). So 1 act as a treshold to select the nunmber of pc

- S: lets consider what 1 is for the correlation matrix above and find an equivalent for S. 1 is also the *average eigenvalue/variance* (sum of eigenvalues is p divided by number of elements p). Therefore we can translate the Kaiser's rule in the non standardized world by keeping as many PCs as are the eigenvalues of S which are geq than the mean variance/eigenvalue

$$\bar{l} = fracsum_{k=1}^{p} l_k p$$

In some books a slightly modified version of the rule above (due to Joliffe) are found:

TODO: RECHECK 08/25/2 sull'idea di Joliffe's

- for R: considering that I'm working in a sample if i reject eigenvalues below 1/mean it may be that i'm refusing something that in the population is above 1/mean and the sample was "particular".
In other words, variability slightly lower than 1 can be kept retained if in the population are actually 1. After som simulation Joliffe ended with choosing to lower the sample treshold to 0.7 (instead of 1)

- in the case of unstandardized data we use $0.7 \cdot \bar{l}$

3.6.3 Screeplot

DO: inserire immagi-

It's a graphical rule common to both matrix S, R. We plot the eigenvalues in order from the highest to the lower to describe the fall/slope.
At some point in the graph there will be an *elbow*: when the lines stop decreasing a lot and becomes almost flat, all the PC passed that point capture the same amount of variability (so we can either take one of them or we throw them all away).
The rule is: keep as many PC as are the eigenvalues *preceding* the elbow point (that is the point from which the screeplot becomes flat so in the example above we should have taken 2 PC)

PC interpretation

How do we interpret principal components?

Assume we have two variables (height and weight) and that the variabiles are mean centered (to ease graph we're working with variances so no probls)

TODO: CHECK 08/25/2

There are two strategies to do it

- one way we do by looking at the extremes (look at coefficients/element of eigenvector). We study sign and the value of the elements of the eigenvectors and try to figure out what happens at the extremes. Important: remember that eigenvector has unit norm: if norm is unit we can measure the importance of each variable for a component and look how variables balance in the vector (this is the preferite way of profs).
;ost of the case first PC have the same sign on all elements of the eigenvector (not always, but often is like that), while the other are contrasts.
- otherwise we look at the correlation between each principal component and the observed varaibles

Extremes interpretation

Strategy is to look at extremes and name the pc acccording to the extremes (reification of PC: give meaning to math construct from an empirical POV).

If we project the point:

- into the first direction we're able to order individuals on the "size" variable: looking at the first component at one end we have mes with high height/weight (BIG men) at the other low height/weight (SMALL men)
- along the second orthogonal direction we have that are characterizing individual according to the "shape": on one hand we have short and fat men, on the other tall but thinny

Example 3.6.1. Before looking at a second criteria of interpreting the PCs we see a popular example with 6 variables on chicken (galline bianche sperone, white leghorn chiken) which are:

- length cranium (head)
- width crannium (head)

- length humerus (wing)
- length ulna (wing)
- lenght femur (legs)
- length of tibia (legs)

with 6 variables we have up to 6 PC: the example is famous because to all 6 components we can give a meaning! (usually is not that so).
Data available is on correlation matrix (we have a $6x6$ matrix only): with R's `eigen` we calculates eigenvalues and vectors eigenvalues:

- the first PC accounts for 76%, while the first two we have more than 80% (so 2 are enough according to proportion of explaiend variance).
- applying kaisers rule we would conclude that 1 PC is enough; to lower the bound accordin to joliff we take two pc (second has eigenvector 0.71)

Our focus is now interpreting: each column correspond to a eigenvector, plotted in the order of the orresponding eigenvalues so vectors matrix is our **A**

TODO: CHECK

1. the first eigenvector is all negative (could be positive either): to change looking at elements of the vector (abs) they are more or less the same (between 0.3 and 0.4): to interpret (assume is positive to ease) which is a chicken with an high value on the first principal compoents look like?

   $$Y_1 \approxeq 0.3(x_1 + x_2) + 0.4(x_3 + x_4 + x_5 + x_3)$$

   a chicken with large first PC1 will be big while small values will corrspond to small chicken; so PC1 describes the size of the chicken.

2. the second is a contrast (not all are the same) otherwise they are not orthogonal: we have positive values with two variables describing the head and negative values for wings and legs. I need a large value of positive: high score on the second are for chicken with large head and small body, low scores small head and big body.

   $$Y_2 \approx 0.6(x_1 + x_2) - 0.2(x_3 + ... + x_6)$$

   so PC2 can be seen as a "shape" (large/small head vs body).
   These are the main two components; most variability of the chicken are related to size and shape

3. if we go on and consider the third PC: values with body characteristics are really near to 0 so I can ignore this, only the head variables is important

   $$y_3 \approx 0.77x_1 - 0.63x_2$$

   I have again a contrast (because of the sign): on one side chicken with long head and not large, on the other side short and large head. So this third PC describes "head shapes"

4. the fourth PC neglect/ignore the head data. We focus on legs and wings

$$y_4 = -0.5(x_3 + x_4) + 0.5(x_5 + x_6)$$

large positive values correspond to chichek with big legs and small wings, on the other side large wings and small legs (another measure of shape of the body)

5. and so on ...

Example 3.6.2. Recent paper (global spectrum of plant forrm and function (sandra diaz: most famous living ecologist argentinian) published in nature main characteristics of plants all around the world.

They were able to define 2 pc starting from 6 variables of plants from all over the world

Variables considered are length plant, stem density (how much production), leaf size/area, leafs mass (how heavy), leaf nitroge content and diaspora mass. 46000 plants

the combinations of 6 features found to be in a two dimensional space which can be plotted

Two principal components were enough and plotted all the plants: heatmap means density and two cluster emerged

the first goes from small plants to very big plants (moving from left to rigth, while the second pc (from sotto to sopra) describes the size of the leaves (from to very broad to very thin

biplot: two variables (original variables and pc) plotted in the same plot

3.6.4 Correlation between PC and variables

Study of the correlation between each principal component and the observed varaibles? We're interested in $cov(y_1, X)$ with y_1 is a scalar random variable (1×1), X is a vector of observed variable ($px1$ vector).
If we assume to work with mean centered observation X, the covariance is

$$\text{Cov}\,(y_1, X) = \mathbb{E}\,[y_1 X] \overset{(1)}{=} \mathbb{E}\left[a_1^T X X\right] \overset{(2)}{=} a_1^T\,\mathbb{E}\left[X X^T\right] = a_1^T \Sigma$$

where in

- we just replaced the first principal component.
- a_1 is a constant vector that can be taken out of the first oiperation

Thus the covaraince between y_1 and X is $a_1^T \Sigma$. But also we have that

$$a_1^T Sigma = lambda a_1 a_1^T$$

Let's see the correlation

$$\text{Corr}\,(y_1, X) = \frac{1}{sd(y_1)} \cdot \text{Cov}\,(y_1, X) \cdot \Delta^{-1/2}$$

at the last passage we needed to derive by the standard deviation of X (usual diagonal matrix Δ). Now we have a problem: we need to define the std of first PC: which is the $sd(y_i)$? it's the $\sqrt{variance} = \sqrt{\lambda_1}$

$$= frac1sqrtlambda_1lambda_1a_1^T Delta^{-1/2}$$
$$= sqrtlambda_1a_1^T Delta^{-1/2}$$

in the sample the formula to adopt is

$$\text{Corr}\,(y_1, X) = \sqrt{l}a_1^T D^{-1/2}$$

TODO: check 09/26

TODO: risentire

corr is a function of vector a_1^T but while elements of eigen are constrained to be unit norm, here we just work with pairwise correlatin. with these way we loose the multivariate feelings ( )

Important remark 11 (PCA and factor analysis). Speaking about factor analysis will see that therer are connection with PC: PCA can be obtained using the same code used for fittining a factor analytics models, by putting some constraints. This is useful from programming pov because with one procedure we have two tools at our disposal. However this caused a lot of confusion, people think that two methods are equivalent (thinking one of them is special case of other) so people used methods from factor analysis on PC but being.
some rotates principal components to improve interpretability (a thing is usually done in factor analyis): this thing drives prof crazy. PC is an axis rotation done in order to maximize the variability explained. We've seen that PC solution is unique (but sign). This means that if we rotate it to improve interpetability we have no more PC (we obtain something that is easier to interpret but no longer the obtained first PCs optimize the explained variability).
if we rotate PC to aid interpretation we no longer have PC

3.7 Principal component scores

Scores are nothing but values of each components y. Dealing with sample data we have Y which is $n \times p$ matrix which contains the coordinates of the n units in the space spanned by the principal components.
In this matrix we have for each unit the element of $\mathbb{R}^p$ from the components
this is obtained as

$$\underset{n\times p}{Y} = \underset{n\times p}{X}\underset{p\times p}{A}$$

where X is original data matrix and A is the matrix of the eigevectors.
if we don't want to keep all the PCs but only m we'll have

$$\underset{n\times m}{Y_m} = \underset{n\times p}{X}\underset{p\times m}{A_m}$$

where A_m is the matrix containing the first m eigenvectors.

3.8 Importance of last components

Now focusing on the last PC is it useful? the last PC have a meaning and there are at least two situations in which last PC is important

3.8.1 Goodness of first PCs

Assume we performed PC in this room two points (projecting students head and lamp, in 3d on the floor in 2d). two dimension are reasonably enough to project (the students) but if we project the lamp too it will be near the students over it is, despite being distant from them in 3d

How can we measure the *quality of representation*: if there's a unit that is very distant (lamp) from the other how can say this

TODO: CHECK 09 26(2

It turns out that the coordinates along the last PC are a measure the quality of the first pcs. I can measure the distance of each units from the projection using only the first m PCs: this below is a measure of the quality of the m dimensional representation of unit j provided by the first m principal components.

$$y_{j,m+1}^2 + y_{j,m+2}^2 + \ldots + y_{j,p}^2$$

If the measure above is much larger than its average, with respect to all the units, then this means that unit j is badly represented by m PCs alone.

Idea is: last PCs cannot be thrown away since they provides us goodness of the first PC.

3.8.2 Multicollinearity

Another possible use of last PC is in multicollinearity diagnosis (for linear regression), other than methods such as VIF.

Multicollinearity means in regression that in the X we have one column is almost a combination of other vectors. If this is so we can use PC to spot it If there are eigenvalue close to 0 there's multicollinearity in data.

TODO: CHECK 09 26

3.9 Orthogonal least square

When performed regression usually we wanted to minimize the sum of square of black lines.

TODO: inserisci immagine

PC was first invented by Pearson in 1901: he wanted to find the line which minimize the sum of the squared *orthogonal distances* (violet) between points and the line. Methods is also known as orthogonal least square.

This was a way to take into account possible measurement errors in the random variable X (while in regression analysis this is considered fixed/without errors)

If we have that: p_j is a point, o is the origin, p'_j is the orthogonal projection of p_j along the red line; i can join p_j the origin obtaining a rectangle triangle so applying pitagora

$$\overline{op_j}^2 = \overline{p_j p'_j}^2 + \overline{op'_j}^2$$

Let's sum with respect to all the units (with j is the generic units)

$$\sum_{j=1}^{n} \overline{op_j}^2 = \sum_{j=1}^{n} \overline{p_j p'_j}^2 + \sum_{j=1}^{n} \overline{op'_j}^2$$

we see the terms

- the overall $TSS = \sum_{j=1}^{n} \overline{op_j}^2$ which is fixed. What is this part: if working with mean centered data is the deviation from the means is the Var $[(] x) * n$: the blue is the total sum of square, which is fixed (and does not depend on the reference system)
- the quantity $\sum_{j=1}^{n} \overline{p_j p_j'}^2$ which is the quantity Pearson wanted to minimise. He wanted to find the line that minize this sum
- the quantity $\sum_{j=1}^{n} \overline{op_j'}^2$, that is TSS along the direction, which it turns out to be maximized in the process. This is the sum of square along the red line (which if the other is minimized, this is maximized). But this is the variability of the first PC component! So solving Pearson's problem amounts to find the line/direction along which the TSS along the direction is maximum and this coincides with solving hotelling's problem.

3.10 PCA and SVD

SVD is a matrix decomposition, way to decompose matrix X, that tells us that given an $n \times p$ matrix X, we can decompose it it in the product of three matrixces

$$\underset{n\times p}{X} = \underset{n\times p}{U} \underset{p\times p}{P} \underset{p\times p}{V}$$

where:

- X is the data matrix
- U is the matix of the left singular vectors (that is the eigenvectors of XX^T corresponding to non 0 eigenvalues). Note that XX^T is not we used so far (X^TX) XX^T is $n \times n$ and has

TODO: CHECK

- P is a square diagonal matrix, whose diagonal entries are the singular values of X, which are the square root of the non zero eigenvalues of XX^T (or X^TX, as the two matrices have the same eigenvalues)
- V is the matrix of right singular vectors (i.e. hte eigenvectors of X^TX)

Now let's assume we work on mean centered data and consider $\tilde{X}$; this means that the codeviance matrix can be written as product of n times the covariance matrix

$$\tilde{\mathbf{X}}^T\tilde{\mathbf{X}} = n\mathbf{S}$$

from matrix algebra it turns out that $\tilde{\mathbf{X}}^T\tilde{\mathbf{X}}$ and S have the same eigenvectors (what changes is only eigenvalues) and the eigenvalues of $\mathbf{S}$ will be the eigenvalues of $\tilde{X}^T\tilde{X}$ divided by n.
We know that eigenvectors of $\mathbf{S}$ are the elements of matrix $\mathbf{A}$ (solution to the PC problems): so A will be the matrix of the eigenvectors of both $\tilde{\mathbf{X}}^T\tilde{\mathbf{X}}$ and $\mathbf{S}$.
If we consider the singular value decomposition of $\tilde{X}$ we have

$$\tilde{\mathbf{X}} = \mathbf{U}\mathbf{P}\mathbf{V}^T$$

and $\mathbf{V} = \mathbf{A}$ Nowwe can replace this. Let's go back to the PC scores Y

TODO: check

$$\mathbf{Y} = \tilde{\mathbf{X}}^T\mathbf{A}$$

we decompose replaceing **X** tilde with its SVD

$$\mathbf{Y} = \mathbf{UPV}^T\mathbf{A} = \mathbf{UVA}^T\mathbf{A}$$

but **A** is orthogonal so $\mathbf{A}^T\mathbf{A} = \mathbf{I}$ so

$$\mathbf{Y} = \mathbf{UP}$$

so using SVD we obtained all the main elemnts of PC analysis **A** (which is V) and UP is the matrix with the PC scores.

Important remark 12. So we can obtain PC by spectral decomposition or by SVD of centered data matrix.

$$\begin{aligned} V &= A \\ P &= n^{1/2}L^{1/2} \\ UP &= Y \end{aligned}$$

from SVD we can obtain PC

moreover if we consider only m principal components then we will have that

$$\tilde{X} \approxeq \mathbf{U}_m\mathbf{P}_m\mathbf{V}_m^T$$

there's a theorem due to eckart and young that states that the equation above is the best rank m approximation of $\tilde{\mathbf{X}}$

So if we have

non other rank m approximation are better than this[2]

TODO: CHECK 09/26

3.11 Lab

We do PCA both manually and results are different from functions. Idea is to find

```
# we should check that variables are correlated


job <- read.table("data/job_perf.txt", header = TRUE)
head(job) # 6 var

##   commun probl_solv logical learn physical appearance
## 1     12         52      20    44       48         16
## 2     12         57      25    45       50         16
## 3     12         54      21    45       50         16
## 4     13         52      21    46       51         17
## 5     14         54      24    46       51         17
## 6     14         48      20    47       51         18
```

[2]svd song it had to be U - the SVD song statistical song youtube michael greenacree

```
## regarding the dataset, look at the pdf with description of dataset
## - commun: communication skill
## - plobl_solv: problem solving

## to perform PCA manually we use matrix comutation

X <- as.matrix(job) # first transform to matrix
# we need number of obs in our dataset
n <- nrow(job)

## in ordere to  perform PCA we need to do spectral value decompositon of
## covariance/correlation matrix

## start with the centering matrix
C <- diag(n) - 1/n * rep(1, n) %*% t(rep(1, n)) # centering matrix
Xc <- C %*% X  #centered data
colMeans(Xc) # check its' approx 0, rounded it's 0

##        commun    probl_solv       logical         learn      physical
## -5.639933e-16  6.679102e-15 -3.792522e-15 -9.414691e-16  1.059597e-14
##    appearance
## -2.646772e-15

## now centered the data compute hte covariance matrix
S <- 1/n * t(Xc) %*% Xc

## now we can also have correlation matrix in order to check if it makes

cor(X) # the quick way, otherway use solve ...

##               commun probl_solv   logical     learn  physical appearance
## commun     1.0000000  0.1313258 0.2252998 0.9821863 0.9767974  0.9692466
## probl_solv 0.1313258  1.0000000 0.3637625 0.1425815 0.1640910  0.1287805
## logical    0.2252998  0.3637625 1.0000000 0.2307109 0.2582155  0.2164945
## learn      0.9821863  0.1425815 0.2307109 1.0000000 0.9814717  0.9718918
## physical   0.9767974  0.1640910 0.2582155 0.9814717 1.0000000  0.9690222
## appearance 0.9692466  0.1287805 0.2164945 0.9718918 0.9690222  1.0000000

## there are some variables highly correlated (learning and commun, phisical
## and comm, ...). It makes sense to perform PC analysis

## we need to chose if work with standardized data or not. To choose it we can
## look at variances of variables. Either using diag(S) or using apply/lapply

apply(X, 2, var)  # check the changes

##     commun probl_solv    logical      learn   physical appearance
##   7.528163   5.810612   6.183265   8.042449   5.810612   8.955510

diag(S)
```

```
##     commun probl_solv    logical      learn   physical appearance
##     7.3776     5.6944     6.0596     7.8816     5.6944     8.7764

# var have the similar variances it's not needed to work with standardized
# data. For this dataset we can work with covariance matrix
## to compute PCA do spectral decomposition (using eigen function)

spectral <- eigen(S)

# obtain a list of two objects: values (eigenvalues), and vectors (matrix of
# eigenvetors)

spectral$values

## [1] 29.7582011  7.4672075  3.7076488  0.2868423  0.1467846  0.1173156

A <- spectral$vectors

## now we can
## 1) look at A this is the loading matrix (rows variables and columns
## principal components, we cana try to interpret)
## 2) look at variablity explained by PC (looking at eigenvalue)

spectral$value/sum(spectral$value) # percentage explained by each single component

## [1] 0.717341652 0.180002109 0.089375394 0.006914529 0.003538342 0.002827973

# first PC explain 72% while the second the 18%, the last for a small
# percentage the cumulative

cumsum(spectral$value/sum(spectral$value))

## [1] 0.7173417 0.8973438 0.9867192 0.9936337 0.9971720 1.0000000

## so we can stop at two components (90% of variability)
## another thing we can do is to extract total variablity explained

sum(spectral$values) # equivalent to sum of variances of our variables

## [1] 41.484

sum(diag(S))  # as predicted

## [1] 41.484

## compute the scores

(Y = scores = X %*%  A)

##             [,1]      [,2]     [,3]      [,4]      [,5]      [,6]
##  [1,] -64.87341 -41.84505 25.32417 -15.39198 -30.05441 -1.714282
##  [2,] -67.36502 -48.66144 25.47772 -16.03400 -31.33531 -2.216466
##  [3,] -66.55825 -43.77018 26.10584 -16.12402 -31.45406 -2.305286
##  [4,] -68.35267 -42.07847 24.73539 -16.23449 -31.55929 -2.434767
##  [5,] -69.42762 -45.48682 24.09698 -16.54983 -30.71906 -2.555887
##  [6,] -69.40433 -38.34110 22.61432 -16.16072 -31.15629 -1.925580
##  [7,] -71.71868 -46.60048 21.35816 -16.60388 -30.75808 -2.628104
##  [8,] -72.11557 -49.37685 22.81622 -16.56140 -30.68727 -2.589067
##  [9,] -71.76839 -44.39077 24.87825 -16.98834 -31.13436 -1.969114
## [10,] -71.85515 -45.08124 25.59607 -16.97946 -31.11609 -1.961633
## [11,] -71.76839 -44.39077 24.87825 -16.98834 -31.13436 -1.969114
## [12,] -72.98581 -47.81286 21.42325 -16.07338 -30.98747 -1.862460
## [13,] -72.65761 -43.49557 25.60775 -16.54909 -30.32761 -2.147892
## [14,] -73.37647 -44.20381 20.68310 -16.73019 -30.87929 -2.774179
## [15,] -73.68645 -44.06550 26.35664 -17.08802 -31.20075 -2.092747
## [16,] -74.22788 -43.99251 24.95557 -17.46169 -30.44731 -2.269498
## [17,] -74.72528 -43.21380 22.84651 -16.62787 -30.40089 -2.244184
## [18,] -74.62554 -43.18480 25.67293 -16.64180 -30.39630 -2.262411
## [19,] -74.81204 -43.90427 23.56433 -16.61899 -30.38261 -2.236703
## [20,] -75.04516 -43.80603 29.22754 -16.86470 -30.93419 -2.943382
## [21,] -76.28765 -47.26506 24.35806 -17.10846 -31.18723 -2.133408
## [22,] -75.97243 -42.28631 24.99832 -17.57912 -30.55025 -2.408093
## [23,] -76.21970 -45.01917 30.69602 -17.55754 -30.47254 -2.396398
## [24,] -77.05322 -45.00343 22.23871 -16.68000 -30.41933 -2.318034
## [25,] -76.60646 -42.21257 22.19387 -16.72946 -30.48785 -2.366184
## [26,] -77.62362 -49.16073 25.13241 -16.61976 -30.31197 -2.264036
## [27,] -77.63646 -46.27505 26.52147 -17.00614 -30.70883 -1.603412
## [28,] -77.56269 -44.92312 22.25941 -17.00996 -30.74998 -1.600146
## [29,] -78.66738 -48.97767 25.17507 -16.11398 -30.57562 -1.522467
## [30,] -77.50059 -41.98601 26.47616 -16.24458 -30.74460 -1.651957
## [31,] -79.55467 -48.17399 23.06655 -16.71438 -30.40122 -2.376922
## [32,] -78.94723 -41.11652 22.97513 -17.16797 -30.90315 -1.779411
## [33,] -79.30724 -43.21692 22.30216 -17.12740 -30.85291 -1.738741
## [34,] -80.08688 -45.93855 25.18380 -17.30972 -31.34057 -2.379042
## [35,] -79.29309 -40.38582 22.26768 -17.39469 -31.48219 -2.457115
## [36,] -79.86426 -40.95904 24.42068 -16.55704 -31.39463 -2.435067
## [37,] -80.75779 -46.54077 24.51037 -16.45814 -31.25760 -2.338766
## [38,] -81.21246 -45.77732 22.39149 -16.84069 -30.52244 -2.522998
## [39,] -80.97609 -45.04335 25.91330 -16.87047 -30.53382 -2.557820
## [40,] -81.17543 -42.87714 25.19376 -17.23404 -30.91700 -1.871488
## [41,] -81.27517 -42.90615 22.36734 -17.22010 -30.92159 -1.853261
## [42,] -81.74659 -43.45036 27.34675 -16.39639 -30.82943 -1.849440
## [43,] -82.58909 -45.52504 25.27094 -16.56669 -31.34226 -2.469880
## [44,] -84.17859 -51.74592 23.27273 -16.77494 -31.52220 -1.629984
## [45,] -83.05660 -41.87591 24.54112 -17.33562 -31.00396 -1.993489
## [46,] -83.45502 -37.48412 24.51778 -15.73833 -30.97274 -2.018351
## [47,] -85.23584 -42.26217 22.49896 -16.23982 -30.66132 -2.776506
## [48,] -86.76324 -45.54595 24.71749 -15.68268 -30.83588 -1.988420
## [49,] -87.68027 -45.38847 26.16304 -15.07176 -31.32736 -2.644076
## [50,] -88.86272 -43.04729 25.46779 -16.19660 -30.19906 -2.329474
```

```
# matrix with same dimenson of our original dataset
# (for each obs the column are the principal components)
## when we select a lower number of components we reduce the number of
# variables

## ----------------------------------------------------------------
## Let's see how this is done using R function prcomp and princomp
## ----------------------------------------------------------------

## prcomp: ?prcomp
# x is our original dataset (matrix or data.frame)
# scale. if we want to work with non stadndardized scale.=FALSE

pca_job <- prcomp(X, scale.=FALSE)
View(pca_job)

## Error in as.data.frame.default(x):  coercizione di classe '"prcomp"'
in data.frame non possibile

## sdev: square root eigenvalue
## rotations: matrix eigenvectors
## center:
##   scale = FALSE
## x matrix of scores

pca_job$sdev^2    # to find eigenvalues

## [1] 30.3655113  7.6195995  3.7833151  0.2926963  0.1497802  0.1197098

spectral$values   # theyre a bit different

## [1] 29.7582011  7.4672075  3.7076488  0.2868423  0.1467846  0.1173156

# this is due to the fact that we computed starting from the biased covariance
# matridx while prcomp use the unbiased covariance matrix
spectral$values*n/(n-1) # exact same results as prcomp

## [1] 30.3655113  7.6195995  3.7833151  0.2926963  0.1497802  0.1197098

## another thing we can do is to use summary:
# obtain std dev, prop variance explained and cumulative,
# obtained previously: same results as before

summary(pca_job)

## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
## Standard deviation     5.5105 2.7604 1.94507 0.54101 0.38701 0.34599
## Proportion of Variance 0.7173 0.1800 0.08938 0.00691 0.00354 0.00283
## Cumulative Proportion  0.7173 0.8973 0.98672 0.99363 0.99717 1.00000

## check rotation/loadings: same results *up to sign changes*. In absolute
## values they're exactly the same

pca_job$rotation
```

```
##                   PC1         PC2         PC3          PC4         PC5
## commun     0.49155807 -0.08748571 -0.01214618  0.380639251 -0.75573635
## probl_solv 0.08675503  0.69046582 -0.71781729 -0.008879732 -0.01827673
## logical    0.13662756  0.70496706  0.69539380 -0.015845890 -0.01597971
## learn      0.50947274 -0.08031174 -0.02069595  0.329958740  0.33064205
## physical   0.43261611 -0.04023081 -0.01036964  0.217845082  0.56076816
## appearance 0.53428266 -0.10274312 -0.02196412 -0.835735944 -0.06699316
##                     PC6
## commun     -0.185864772
## probl_solv  0.007480667
## logical     0.016594522
## learn       0.717887386
## physical   -0.670223722
## appearance  0.023681484
```

```
A
```

```
##             [,1]        [,2]        [,3]         [,4]        [,5]         [,6]
## [1,] -0.49155807  0.08748571  0.01214618 -0.380639251  0.75573635 -0.185864772
## [2,] -0.08675503 -0.69046582  0.71781729  0.008879732  0.01827673  0.007480667
## [3,] -0.13662756 -0.70496706 -0.69539380  0.015845890  0.01597971  0.016594522
## [4,] -0.50947274  0.08031174  0.02069595 -0.329958740 -0.33064205  0.717887386
## [5,] -0.43261611  0.04023081  0.01036964 -0.217845082 -0.56076816 -0.670223722
## [6,] -0.53428266  0.10274312  0.02196412  0.835735944  0.06699316  0.023681484
```

```
## we can extract teh matrix containing the scores
pca_job$x
```

```
##               PC1         PC2         PC3         PC4          PC5         PC6
##  [1,] -12.09655785 -2.55639519 -0.87102139 -1.26439296 -0.795479934  0.47326072
##  [2,]  -9.60493991  4.25999586 -1.02457409 -0.62237217  0.485416246 -0.02892339
##  [3,] -10.41171526 -0.63126985 -1.65269740 -0.53234941  0.604165275 -0.11774348
##  [4,]  -8.61729573 -2.32297288 -0.28223872 -0.42188282  0.709399435 -0.24722444
##  [5,]  -7.54234490  1.08537423  0.35616191 -0.10654070 -0.130829504 -0.36834431
##  [6,]  -7.56562995 -6.06034380  1.83883039 -0.49565596  0.306398598  0.26196247
##  [7,]  -5.25128766  2.19903820  3.09498469 -0.05249151 -0.091811207 -0.44056103
##  [8,]  -4.85439500  4.97540273  1.63692662 -0.09497660 -0.162621102 -0.40152451
##  [9,]  -5.20157015 -0.01067595 -0.42510373  0.33197106  0.284472961  0.21842893
## [10,]  -5.11481512  0.67978988 -1.14292102  0.32309132  0.266196234  0.22590960
## [11,]  -5.20157015 -0.01067595 -0.42510373  0.33197106  0.284472961  0.21842893
## [12,]  -3.98414966  3.41141622  3.02990113 -0.58299434  0.137581243  0.32508302
## [13,]  -4.31235698 -0.90587185 -1.15460783 -0.10727975 -0.522276836  0.03965112
## [14,]  -3.59349570 -0.19763189  3.77004958  0.07382097  0.029402664 -0.58663652
## [15,]  -3.28351310 -0.33594857 -1.90349072  0.43164434  0.350856649  0.09479545
## [16,]  -2.74208249 -0.40893305 -0.50242581  0.80531744 -0.402582683 -0.08195547
## [17,]  -2.24468232 -1.18764076  1.60663845 -0.02850493 -0.449002100 -0.05664080
## [18,]  -2.34442739 -1.21664323 -1.21978373 -0.01457262 -0.453596132 -0.07486851
## [19,]  -2.15792729 -0.49717494  0.88882115 -0.03738467 -0.467278828 -0.04916013
## [20,]  -1.92480131 -0.59541069 -4.77439284  0.20832505  0.084301271 -0.75583927
## [21,]  -0.68231822  2.86361410  0.09509129  0.45208818  0.337341779  0.05413458
## [22,]  -0.99753550 -2.11513731 -0.54517821  0.92275019 -0.299645540 -0.22055028
## [23,]  -0.75026044  0.61772351 -6.24286955  0.90116358 -0.377346483 -0.20885532
## [24,]   0.08325742  0.60198780  2.21443285  0.02363068 -0.430557547 -0.13049071
## [25,]  -0.36350777 -2.18887797  2.25927984  0.07308193 -0.362044668 -0.17864109
## [26,]   0.65366014  4.75928397 -0.67925980 -0.03661387 -0.537920897 -0.07649285
## [27,]   0.66649515  1.87360524 -2.06831985  0.34976228 -0.141062985  0.58413030
## [28,]   0.59273016  0.52167606  2.19373690  0.35358942 -0.099915497  0.58739668
## [29,]   1.69741554  4.57622911 -0.72191988 -0.54239107 -0.274272008  0.66507602
## [30,]   0.53063003 -2.41543654 -2.02301349 -0.41179680 -0.105286827  0.53558622
## [31,]   2.58470723  3.77254800  1.38659268  0.05800684 -0.448666449 -0.18937929
## [32,]   1.97726699 -3.28492815  1.47801420  0.51159369  0.053257797  0.40813215
## [33,]   2.33727715 -1.18452821  2.15098450  0.47102217  0.003021646  0.44880186
## [34,]   3.11691339  1.53710428 -0.73065430  0.65334833  0.490682899 -0.19149919
## [35,]   2.32312807 -4.01562478  2.18546185  0.73831850  0.632302688 -0.26957224
## [36,]   2.89429323 -3.44240332  0.03246935 -0.09933102  0.544735784 -0.24752394
## [37,]   3.78782361  2.13932821 -0.05722463 -0.19823351  0.407710027 -0.15122318
## [38,]   4.24249919  1.37587791  2.06165757  0.18431932 -0.327452578 -0.33545477
## [39,]   4.00612655  0.64190838 -1.46015840  0.21409753 -0.316066899 -0.37027700
## [40,]   4.20546167 -1.52430206 -0.74061357  0.57766162  0.067108318  0.31605453
## [41,]   4.30520673 -1.49529959  2.08580861  0.56372930  0.071702350  0.33428224
## [42,]   4.77662682 -0.95108060 -2.89360607 -0.25998790 -0.020458586  0.33810282
## [43,]   5.61912563  1.12358977 -0.81779432 -0.08968049  0.492370442 -0.28233733
## [44,]   7.20862885  7.34447785  1.18042064  0.11856372  0.672304039  0.55755919
## [45,]   6.08663622 -2.52553927 -0.08797217  0.67924848  0.154065750  0.19405424
## [46,]   6.48505376 -6.91732417 -0.06462994 -0.91804654  0.122848747  0.16919164
## [47,]   8.26587583 -2.13927375  1.95418864 -0.41655112 -0.188571452 -0.58896291
## [48,]   9.79327892  1.14450174 -0.26434680 -0.97369314 -0.014009185  0.19912315
## [49,]  10.71030516  0.98702657 -1.70989165 -1.58461785  0.477468801 -0.45653294
## [50,]  11.89275641 -1.35415530 -1.01463918 -0.45977525 -0.650828679 -0.14193096
```

```
head(scores) # check first
```

```
##            [,1]      [,2]     [,3]      [,4]      [,5]      [,6]
## [1,] -64.87341 -41.84505 25.32417 -15.39198 -30.05441 -1.714282
## [2,] -67.36502 -48.66144 25.47772 -16.03400 -31.33531 -2.216466
## [3,] -66.55825 -43.77018 26.10584 -16.12402 -31.45406 -2.305286
## [4,] -68.35267 -42.07847 24.73539 -16.23449 -31.55929 -2.434767
## [5,] -69.42762 -45.48682 24.09698 -16.54983 -30.71906 -2.555887
## [6,] -69.40433 -38.34110 22.61432 -16.16072 -31.15629 -1.925580
```

```
#htey are really different: prcomp by default centers the data (when we did it
#manually we didn't centere the data). We can

pca_job$center # this values? check
```

```
##     commun probl_solv    logical      learn   physical appearance
##      17.68      54.16      24.02      50.28      54.16      21.06
```

```
# if we don't want to cenceter the data set center=FALSE in call
```

```
head(Xc %*% A) # if we center the data when computing manually we get same results

##            [,1]       [,2]       [,3]      [,4]       [,5]        [,6]
## [1,] 12.096558  2.5563952  0.8710214 1.2643930  0.7954799  0.47326072
## [2,]  9.604940 -4.2599959  1.0245741 0.6223722 -0.4854162 -0.02892339
## [3,] 10.411715  0.6312698  1.6526974 0.5323494 -0.6041653 -0.11774348
## [4,]  8.617296  2.3229729  0.2822387 0.4218828 -0.7093994 -0.24722444
## [5,]  7.542345 -1.0853742 -0.3561619 0.1065407  0.1308295 -0.36834431
## [6,]  7.565630  6.0603438 -1.8388304 0.4956560 -0.3063986  0.26196247

head(pca_job$x) #

##              PC1        PC2        PC3        PC4        PC5         PC6
## [1,] -12.096558 -2.5563952 -0.8710214 -1.2643930 -0.7954799  0.47326072
## [2,]  -9.604940  4.2599959 -1.0245741 -0.6223722  0.4854162 -0.02892339
## [3,] -10.411715 -0.6312698 -1.6526974 -0.5323494  0.6041653 -0.11774348
## [4,]  -8.617296 -2.3229729 -0.2822387 -0.4218828  0.7093994 -0.24722444
## [5,]  -7.542345  1.0853742  0.3561619 -0.1065407 -0.1308295 -0.36834431
## [6,]  -7.565630 -6.0603438  1.8388304 -0.4956560  0.3063986  0.26196247

# so prcomp use unbiased stdev and center the data by def

## ------------------------------------------------------------------------
## princomp
## ------------------------------------------------------------------------
## the differences are insight (in the way they perform PC): prcomp uses SVD,
## while princomp use eigen

pca_job <- princomp(X, cor = FALSE)

# cor=TRUE to standardize the data (use correlation not covariance
# matrix),=FALSE

## furthermore, we have to specify scores=TRUE (btw default)

summary(pca_job) # same stuff as before

## Importance of components:
##                           Comp.1    Comp.2     Comp.3      Comp.4      Comp.5
## Standard deviation     5.4551078 2.7326192 1.92552560 0.535576640 0.383124752
## Proportion of Variance 0.7173417 0.1800021 0.08937539 0.006914529 0.003538342
## Cumulative Proportion  0.7173417 0.8973438 0.98671916 0.993633685 0.997172027
##                             Comp.6
## Standard deviation     0.342513705
## Proportion of Variance 0.002827973
## Cumulative Proportion  1.000000000

## extract the eigenvaluesù

pca_job$sdev^2  # same as manually
```

```
##     Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6
## 29.7582011  7.4672075  3.7076488  0.2868423  0.1467846  0.1173156
```

```
spectral$values #  (princomp uses biased version of covariance)
```

```
## [1] 29.7582011  7.4672075  3.7076488  0.2868423  0.1467846  0.1173156
```

```
# matrix of loadings (A). there are some empty entries;
# loadings in absolute value < than 0.1 are omitted

pca_job$loadings # but they can be printed the brute force way
```

```
##
## Loadings:
##            Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
## commun      0.492                0.381  0.756  0.186
## probl_solv        -0.690  0.718
## logical     0.137 -0.705 -0.695
## learn       0.509                0.330 -0.331 -0.718
## physical    0.433                0.218 -0.561  0.670
## appearance  0.534  0.103        -0.836
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.167  0.167  0.167  0.167  0.167  0.167
## Cumulative Var  0.167  0.333  0.500  0.667  0.833  1.000
```

```
spectral$vectors #
```

```
##             [,1]        [,2]        [,3]         [,4]        [,5]         [,6]
## [1,] -0.49155807  0.08748571  0.01214618 -0.380639251  0.75573635 -0.185864772
## [2,] -0.08675503 -0.69046582  0.71781729  0.008879732  0.01827673  0.007480667
## [3,] -0.13662756 -0.70496706 -0.69539380  0.015845890  0.01597971  0.016594522
## [4,] -0.50947274  0.08031174  0.02069595 -0.329958740 -0.33064205  0.717887386
## [5,] -0.43261611  0.04023081  0.01036964 -0.217845082 -0.56076816 -0.670223722
## [6,] -0.53428266  0.10274312  0.02196412  0.835735944  0.06699316  0.023681484
```

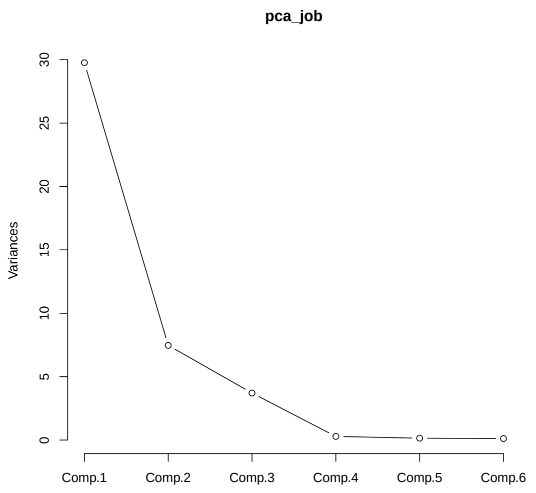
```
pca_job$scores[1:5, ]
```

```
##          Comp.1     Comp.2     Comp.3     Comp.4     Comp.5      Comp.6
## [1,] -12.096558  2.5563952  0.8710214 -1.2643930  0.7954799 -0.47326072
## [2,]  -9.604940 -4.2599959  1.0245741 -0.6223722 -0.4854162  0.02892339
## [3,] -10.411715  0.6312698  1.6526974 -0.5323494 -0.6041653  0.11774348
## [4,]  -8.617296  2.3229729  0.2822387 -0.4218828 -0.7093994  0.24722444
## [5,]  -7.542345 -1.0853742 -0.3561619 -0.1065407  0.1308295  0.36834431
```

```
scores[1:5,]  # diff because princomp centers the data
```

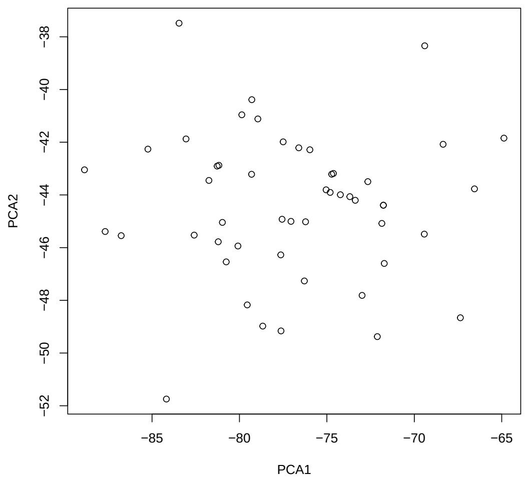
```
##           [,1]      [,2]     [,3]      [,4]      [,5]      [,6]
## [1,] -64.87341 -41.84505 25.32417 -15.39198 -30.05441 -1.714282
## [2,] -67.36502 -48.66144 25.47772 -16.03400 -31.33531 -2.216466
## [3,] -66.55825 -43.77018 26.10584 -16.12402 -31.45406 -2.305286
## [4,] -68.35267 -42.07847 24.73539 -16.23449 -31.55929 -2.434767
## [5,] -69.42762 -45.48682 24.09698 -16.54983 -30.71906 -2.555887
```

```
(Xc %*% A)[1:5, ]

##           [,1]       [,2]       [,3]      [,4]       [,5]        [,6]
## [1,] 12.096558  2.5563952  0.8710214 1.2643930  0.7954799  0.47326072
## [2,]  9.604940 -4.2599959  1.0245741 0.6223722 -0.4854162 -0.02892339
## [3,] 10.411715  0.6312698  1.6526974 0.5323494 -0.6041653 -0.11774348
## [4,]  8.617296  2.3229729  0.2822387 0.4218828 -0.7093994 -0.24722444
## [5,]  7.542345 -1.0853742 -0.3561619 0.1065407  0.1308295 -0.36834431

## check

## ---------------------------------------------------------------------------------
## Selection of n of components
## ---------------------------------------------------------------------------------

# 1. keep as many as neede to explain 80%  of total variabilituy
summary(pca_job) # look at cumulative prop and first two are enough

## Importance of components:
##                           Comp.1    Comp.2     Comp.3      Comp.4      Comp.5
## Standard deviation     5.4551078 2.7326192 1.92552560 0.535576640 0.383124752
## Proportion of Variance 0.7173417 0.1800021 0.08937539 0.006914529 0.003538342
## Cumulative Proportion  0.7173417 0.8973438 0.98671916 0.993633685 0.997172027
##                             Comp.6
## Standard deviation     0.342513705
## Proportion of Variance 0.002827973
## Cumulative Proportion  1.000000000

# 2. screeplot: plot of eigen values against n of components
plot(pca_job, type = 'line') # on x # of components while y the variances (eigenvalue
```



```
# according to this criterion we search for the elbow. In this case we should
# keep 3 components because comp4, 5, 6 are flat
spectral$values # last three are close to zero

## [1] 29.7582011  7.4672075  3.7076488  0.2868423  0.1467846  0.1173156

# 3. kaiser rule: number of components to select equal to pc with variance
# larger than the average variance

pca_job$sdev^2 > mean(pca_job$sdev^2) # the first two component are selected

## Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
##   TRUE   TRUE  FALSE  FALSE  FALSE  FALSE

## ---------------------------------------------------------------------------------------
## Interpretation
## --------------------------------------------------------------------------------------

## 1) if we choose to work with 2 PC most chosen we can try to interpret: focus on
## loadings
pca_job$loadings[, 1:2] # matrix eigenvectors, first two columns

##                 Comp.1      Comp.2
## commun     0.49155807  0.08748571
## probl_solv 0.08675503 -0.69046582
## logical    0.13662756 -0.70496706
## learn      0.50947274  0.08031174
## physical   0.43261611  0.04023081
## appearance 0.53428266  0.10274312

## first: CHECK
## second one has almost all the values close to zero with exception of problem
## solving and logical skill: second seems to discriminate those who have
## logical skill from the other

## 2) lokking at correlation between y_i and X
## corr(y_1, x) = l_1^{1/2} a_1^T D^{-1/2}
## (corresponsing sample expression is at page 12 di PCA1.pdf)
## we need to compute D

D <- diag(diag(S))

# for the first principal compoennt
spectral$values[1]^0.5 * t(A[, 1])   %*% solve(D^0.5)

##            [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
## [1,] -0.9872352 -0.1983234 -0.3027748 -0.9899587 -0.9889676 -0.9838213

# l1                          a_1^T      D^{-1/2}

# either the following is the samne
cor(scores[, 1], X) #
```

```
##           commun probl_solv    logical      learn   physical appearance
## [1,] -0.9872352 -0.1983234 -0.3027748 -0.9899587 -0.9889676 -0.9838213

# the first component is an average, high correlation with other

# for the second principal compoennt
cor(scores[, 2], X) #

##           commun probl_solv   logical      learn   physical appearance
## [1,] 0.08801541 -0.7906737 -0.782575 0.07817195 0.04606954 0.09477061

# correlation low with expect second and third one which have high negative
# correlation

## ----------------------
## visualization of PCs (since we selected 2 components)
## ----------------------

plot(scores[, 1], scores[ , 2], xlab = "PCA1", ylab = "PCA2")
```



```
## ----------------------
## well representation
## ----------------------

# how well units are represented along these components? a way to do this is to
# look at the discarded principal components. we cans see the scores of these
# components for each observation (high wors represented, low good represented)

# sum of squares of discarded components for each individual
# sum of squares by row
(d <- apply(scores[, 3:6], 1, function(x) sum(x^2)))
```

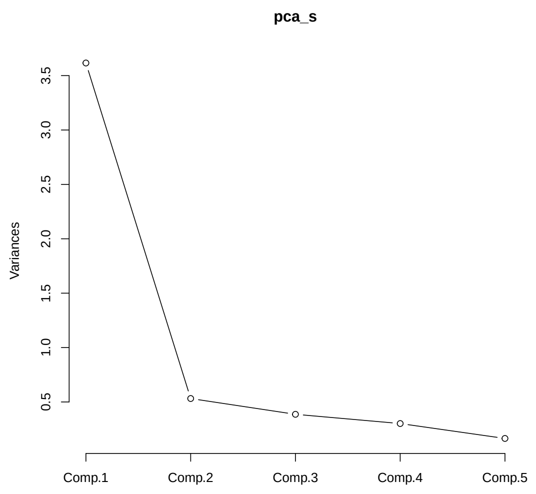
```
##  [1] 1784.433 1893.018 1936.171 1877.315 1804.755 1746.998 1684.826 1743.272
##  [9] 1880.757 1915.520 1880.757 1681.001 1854.007 1668.917 1964.539 1859.880
## [17] 1727.699 1865.102 1759.574 2104.255 1863.209 1873.058 2184.831 1703.492
## [25] 1707.550 1831.796 1938.200 1732.942 1830.631 1912.833 1741.321 1780.767
## [33] 1745.660 1921.742 1795.591 1862.057 1854.136 1722.972 1894.969 1891.101
## [41] 1756.409 1970.561 1901.513 1819.324 1868.010 1812.201 1717.760 1811.706
## [49] 1900.057 1828.348

## if we look at d for each observation: which are the observation represented
## the worst with 2 PCA? order the vector decresingly

(d_order <- order(d, decreasing=TRUE))

##  [1] 23 20 42 15 27  3 34 10 30 43 49 39  2 40  9 11  4 22 45 18 21 36 16 37 13
## [26] 26 29 50 44 46 48  5 35  1 32 19 41  6 33  8 31 28 17 38 47 25 24  7 12 14

# indexes: obs 23 is the worst hwhile 14 is the best represented
```

Now we do the same for the sparrow dataset

```
sparrows <- read.table("data/sparrows.dat", header = TRUE)
head(sparrows)

##   totL AlarE  bhL   hL   kL
## 1  156   245 31.6 18.5 20.5
## 2  154   240 30.4 17.9 19.6
## 3  153   240 31.0 18.4 20.6
## 4  153   236 30.9 17.7 20.2
## 5  155   243 31.5 18.6 20.3
## 6  163   247 32.0 19.0 20.9

# see if PC is justified and if to work with standardized data

(R <- cor(sparrows)) # correlations are high so it makes sense

##            totL     AlarE       bhL        hL        kL
## totL  1.0000000 0.7349642 0.6618119 0.6452841 0.6051247
## AlarE 0.7349642 1.0000000 0.6737411 0.7685087 0.5290138
## bhL   0.6618119 0.6737411 1.0000000 0.7631899 0.5262701
## hL    0.6452841 0.7685087 0.7631899 1.0000000 0.6066493
## kL    0.6051247 0.5290138 0.5262701 0.6066493 1.0000000

# decide to work with covariance or correlation
sapply(sparrows, var)

##       totL      AlarE        bhL          hL          kL
## 13.3537415 25.6828231  0.6316327  0.3184184  0.9828231

## we have different magnitudes of variances: the first two variables takes
## values that are large, while last three have different units of measuremente.
## so we standardize data/use correlation matrix
## Function are as before

spectral <- eigen(R)
spectral$values # eigevalues

## [1] 3.6159783 0.5315041 0.3864245 0.3015655 0.1645275

(A <- spectral$vectors) # eigevalues

##            [,1]        [,2]       [,3]        [,4]       [,5]
## [1,] -0.4517989  0.05072137  0.6904702  0.42041399 -0.3739091
## [2,] -0.4616809 -0.29956355  0.3405484 -0.54786307  0.5300805
## [3,] -0.4505416 -0.32457242 -0.4544927  0.60629605  0.3427923
## [4,] -0.4707389 -0.18468403 -0.4109350 -0.38827811 -0.6516665
## [5,] -0.3976754  0.87648935 -0.1784558 -0.06887199  0.1924341

spectral$values / sum(spectral$values) # prop explained

## [1] 0.72319567 0.10630082 0.07728491 0.06031310 0.03290550

cumsum(spectral$values) / sum(spectral$values) # first two components are enough

## [1] 0.7231957 0.8294965 0.9067814 0.9670945 1.0000000

sum(spectral$values) # the number of variables (working with std data the

## [1] 5

# varainces is 1)

sum(diag(R))

## [1] 5

## compute the scores
X = as.matrix(sparrows)
scores <- X %*% A

# to get the same results of princomp we need to standardize the data
scores_stand <- scale(X)%*% A    # to do it quickly scale(X) performs Z

# compoare
pca_s <- princomp(X, cor = TRUE, scores=TRUE) # cor=TRUE, so princpomp know it has to

pca_s$sdev^2    # are the

##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5
## 3.6159783 0.5315041 0.3864245 0.3015655 0.1645275

spectral$values #  same

## [1] 3.6159783 0.5315041 0.3864245 0.3015655 0.1645275
```

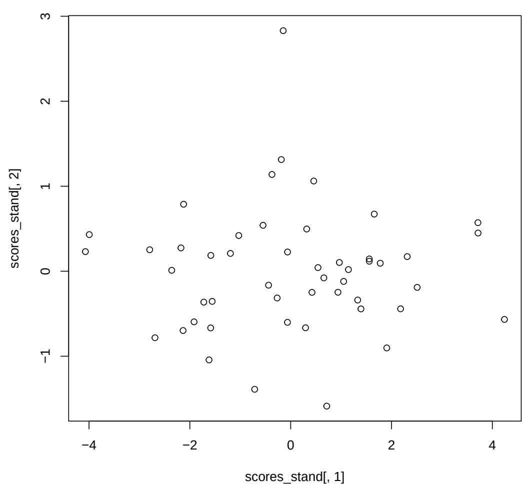
```
pca_s$loadings   # same up to sign

##
## Loadings:
##       Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## totL   0.452         0.690  0.420  0.374
## AlarE  0.462 -0.300  0.341 -0.548 -0.530
## bhL    0.451 -0.325 -0.454  0.606 -0.343
## hL     0.471 -0.185 -0.411 -0.388  0.652
## kL     0.398  0.876 -0.178        -0.192
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## SS loadings       1.0    1.0    1.0    1.0    1.0
## Proportion Var    0.2    0.2    0.2    0.2    0.2
## Cumulative Var    0.2    0.4    0.6    0.8    1.0

spectral$vectors # change (see the uper matrix)

##            [,1]        [,2]       [,3]        [,4]       [,5]
## [1,] -0.4517989  0.05072137  0.6904702  0.42041399 -0.3739091
## [2,] -0.4616809 -0.29956355  0.3405484 -0.54786307  0.5300805
## [3,] -0.4505416 -0.32457242 -0.4544927  0.60629605  0.3427923
## [4,] -0.4707389 -0.18468403 -0.4109350 -0.38827811 -0.6516665
## [5,] -0.3976754  0.87648935 -0.1784558 -0.06887199  0.1924341

## prop variance explained
summary(pca_s)

## Importance of components:
##                           Comp.1    Comp.2     Comp.3    Comp.4    Comp.5
## Standard deviation     1.9015726 0.7290433 0.62163056 0.5491498 0.4056199
## Proportion of Variance 0.7231957 0.1063008 0.07728491 0.0603131 0.0329055
## Cumulative Proportion  0.7231957 0.8294965 0.90678139 0.9670945 1.0000000

head(pca_s$scores) # to compare with what done manually

##           Comp.1      Comp.2     Comp.3       Comp.4     Comp.5
## [1,]  0.06495523 -0.60706359 -0.1730078 -0.521171046 -0.5544775
## [2,] -2.20290734 -0.44688437  0.4042155 -0.652148842 -0.2334712
## [3,] -1.15743714  0.01945365 -0.6831336 -0.723721141 -0.2110360
## [4,] -2.33501516  0.17377503 -0.3091328  0.150836370 -0.4830580
## [5,] -0.29809954 -0.67210136 -0.4791281 -0.551518863 -0.2470115
## [6,]  1.93612015 -0.60142305  0.6273677  0.006677154  0.2884754

scores[1:5, ] # witrhout standardizeing data theyre different

##           [,1]      [,2]     [,3]      [,4]     [,5]
## [1,] -214.6906 -61.18565 165.5251 -58.07794 74.26120
## [2,] -210.2976 -60.07782 163.3940 -56.61205 72.16507
## [3,] -210.7491 -59.53913 162.0469 -56.93170 72.61126
## [4,] -208.3688 -58.52974 161.0892 -54.50153 70.83585
## [5,] -213.2379 -60.79855 164.1936 -57.48831 73.43701
```

```
scores_stand[1:5, ] # they're similar but not equivalent: the reason is that

##               [,1]        [,2]       [,3]       [,4]      [,5]
## [1,] -0.06428901 -0.60083713 -0.1712334 -0.5158256 0.5487904
## [2,]  2.18031283 -0.44230082  0.4000696 -0.6454600 0.2310766
## [3,]  1.14556567  0.01925412 -0.6761269 -0.7162982 0.2088714
## [4,]  2.31106565  0.17199267 -0.3059621  0.1492893 0.4781034
## [5,]  0.29504203 -0.66520783 -0.4742138 -0.5458621 0.2444780

# scale use the unbiased covariance matrix, while princomp uses the biased ones
# to get the exactly used with princpom

n <- nrow(sparrows)
C <- diag(n) - 1/n * rep(1, n) %*% t(rep(1, n)) # centering matrix
Xc <- C %*% X  #centered data
S <- 1/n * t(Xc) %*% Xc
D = diag(diag(S))
Z <- Xc %*% solve(D^0.5)

scores_z <- Z%*% A
scores_z[1:5, ] # now exactly equivalent up to sign changes to

##               [,1]        [,2]       [,3]       [,4]      [,5]
## [1,] -0.06495523 -0.60706359 -0.1730078 -0.5211710 0.5544775
## [2,]  2.20290734 -0.44688437  0.4042155 -0.6521488 0.2334712
## [3,]  1.15743714  0.01945365 -0.6831336 -0.7237211 0.2110360
## [4,]  2.33501516  0.17377503 -0.3091328  0.1508364 0.4830580
## [5,]  0.29809954 -0.67210136 -0.4791281 -0.5515189 0.2470115

pca_s$scores[1:5, ]

##            Comp.1      Comp.2     Comp.3     Comp.4     Comp.5
## [1,]  0.06495523 -0.60706359 -0.1730078 -0.5211710 -0.5544775
## [2,] -2.20290734 -0.44688437  0.4042155 -0.6521488 -0.2334712
## [3,] -1.15743714  0.01945365 -0.6831336 -0.7237211 -0.2110360
## [4,] -2.33501516  0.17377503 -0.3091328  0.1508364 -0.4830580
## [5,] -0.29809954 -0.67210136 -0.4791281 -0.5515189 -0.2470115

## ---------------------
## Select n of components
## ---------------------
summary(pca_s) # two principal components

## Importance of components:
##                           Comp.1    Comp.2     Comp.3    Comp.4    Comp.5
## Standard deviation     1.9015726 0.7290433 0.62163056 0.5491498 0.4056199
## Proportion of Variance 0.7231957 0.1063008 0.07728491 0.0603131 0.0329055
## Cumulative Proportion  0.7231957 0.8294965 0.90678139 0.9670945 1.0000000

plot(pca_s, type="l") # screeplot here seems that two components are enough
```



```
pca_s$sdev^2 >mean(pca_s$sdev^2) # just the first

## Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
##   TRUE  FALSE  FALSE  FALSE  FALSE

# --------------
# interpretation
# --------------
pca_s$loadings[ , 1:2]

##          Comp.1      Comp.2
## totL  0.4517989  0.05072137
## AlarE 0.4616809 -0.29956355
## bhL   0.4505416 -0.32457242
## hL    0.4707389 -0.18468403
## kL    0.3976754  0.87648935

# the first pc has large values of all the variables so its an average measure
# the second is a contrast (kL very high positive, other AlarE to hL are
# negative for alarextent, ... while positive positive for keel: this is a
# measure of the shape (divide animals with small AlarE to )
# ce lo scrive

## either using the correlation
## corr(y) same as before but D now containd diagonal of R not of S

## first component
spectral$values[1]^0.5  *  t(A[, 1]) %*% solve(diag(diag(R))^0.5)

##              [,1]       [,2]       [,3]       [,4]       [,5]
## [1,] -0.8591285 -0.8779197 -0.8567376 -0.8951441 -0.7562086

cor(scores_stand[, 1], X) # same result
```

```
##              totL      AlarE        bhL         hL         kL
## [1,] -0.8591285 -0.8779197 -0.8567376 -0.8951441 -0.7562086

## second component
spectral$values[2]^0.5  *  t(A[, 2]) %*% solve(diag(diag(R))^0.5)

##              [,1]       [,2]       [,3]       [,4]      [,5]
## [1,] 0.03697807 -0.2183948 -0.2366273 -0.1346426 0.6389987

cor(scores_stand[, 2], X) # same result

##              totL      AlarE        bhL         hL        kL
## [1,] 0.03697807 -0.2183948 -0.2366273 -0.1346426 0.6389987

# --------------
# if we consider the first two PC we can perform bidimensional plot of
# --------------

plot(scores_stand[, 1], scores_stand[, 2])
```



```
## ---------------------------
## check if two pc well represent obs
## --------------------------------
d <- apply(scores_stand[, -(1:2)], 1, function(x) {sum(x^2)})
d

##  [1] 0.59656782 0.63007062 1.01385789 0.34448301 0.58261368 0.46712122
##  [7] 0.29466360 3.69812707 0.35484002 0.74873540 0.12956580 0.77957053
## [13] 0.21760626 0.56499691 0.99346361 0.04093662 0.30300244 0.77763915
## [19] 1.27141495 0.86832735 1.00297894 0.50507128 0.20563631 0.95941244
## [25] 0.30766169 1.38839769 0.54485452 2.03454814 1.46970953 0.17835616
## [31] 1.46648326 0.45001132 0.32929718 2.62044731 1.15248224 1.23872269
## [37] 0.48661370 0.82446792 0.14565941 0.81254552 2.62851160 0.68244677
## [43] 0.56803687 0.14585928 0.19455396 0.49686904 1.54877597 0.97997218
## [49] 0.87485689
```

```
(d_order <- order(d, decreasing = T))
```

```
##  [1]  8 41 34 28 47 29 31 26 19 36 35  3 21 15 48 24 49 20 38 40 12 18 10 42  2
## [26]  1  5 43 14 27 22 46 37  6 32  9  4 33 25 17  7 13 23 45 30 44 39 11 16
```

```
sort(d, decreasing =TRUE) ## two discarded pc are noth veri high
```

```
##  [1] 3.69812707 2.62851160 2.62044731 2.03454814 1.54877597 1.46970953
##  [7] 1.46648326 1.38839769 1.27141495 1.23872269 1.15248224 1.01385789
## [13] 1.00297894 0.99346361 0.97997218 0.95941244 0.87485689 0.86832735
## [19] 0.82446792 0.81254552 0.77957053 0.77763915 0.74873540 0.68244677
## [25] 0.63007062 0.59656782 0.58261368 0.56803687 0.56499691 0.54485452
## [31] 0.50507128 0.49686904 0.48661370 0.46712122 0.45001132 0.35484002
## [37] 0.34448301 0.32929718 0.30766169 0.30300244 0.29466360 0.21760626
## [43] 0.20563631 0.19455396 0.17835616 0.14585928 0.14565941 0.12956580
## [49] 0.04093662
```

Capitolo 4

Factor analysis

4.1 Introduction

Thus method was invented by Spearman (psycometrician) in 1904 to measure intelligence (Galton wanted to study correlation of intelligence between father and sons; but measure of intelligence was not available so he measured height, something measurable).

Intelligence is a latent variable: it's not observed (as all psychological variables are).

Spearman had this dataset on students on performance of students in three subjects' test: X_1 classics, X_2 french and X_3 english with correlation matrix

$$\rho = \begin{bmatrix} 1 & 0.83 & 0.78 \\ & 1 & 0.67 \\ & & 1 \end{bmatrix}$$

Variables are positive correlated (and pretty high correlation).

- I observe positive and high correlation: does it mean that these variables are correlated because of another varaible which drives the correlation?
- if i condition /eliminate on the unobserved variable can I diminish the correlation of the observed (in this case the correlation is due to the hidden variable). It's the same of partial corrrelation but with a variable which is not observed

Two appreciable differences of factor analysis with respect to PCA:

- what spearman did was to figure out *a model* which could explain the observed correlation: any model require *assumptions* (while in PCA we had no theoretical assumptions);
- the focus here is to explain *correlation* (off diagonal elements of variance/covariance/correlation matrix), while in PCA was on variance (in diagonal elements)

So is there a latent variable that allows to explain the observed correlation? Spearman assumed that the performance (random variables on single students) is composed as

$$X_1 = \lambda_1 f + u_1$$

where[1]:

- X_1, X_2 and X_3 are the *observed variables* (grades).
- λ_1, λ_2 and λ_3 are known as *factor loadings*
- the quantity f has no subscript so it is the same for all the variables: its called *common factor*
- the $u_1, \ldots, u_3$ are different for each variables and are known as *unique factors*

The idea that Spearman had was:

- I have a given level of intelligence f (which is a random variable)
- my result in classics (X_1) is due to my intelligence (weighed by a contribution λ_1: λ_1 is fixed, not related to the single unit
- then there is u_1, a random variable (like f) which takes into account all other factors impacting performance in classics other than intelligence. This is like ε term in regression: it's random variablity around the model.

Some considerations:

- If data have been generated in this way I can explain the correlation between observed variales ($X_1, \ldots, X_3$):
- this model above looks like a regression model, but is different becaus all what we see on the right side of $=$ is unknown while in regression is all available

After posing the model, spearman research opened big debate: people started to put question regarding intelligence: is there a unique intelligence? actually no, there are different side of intelligence, which is a good motivating example for introducing the the general model of intelligence, a generalization of the one above, applying a general linear factor model, in which a generic test X_i can be explained in terms of m intelligence components

$$X_i = \lambda_{i1} f_1 + \lambda_{i2} f_2 + \ldots + \lambda_{ik} f_k + \ldots \lambda_{im} f_m + u_i$$

where

- we have many f for differnt kind of intelligence
- the weights/loadings lambdas has two subscript, the first concerning the explained/observed variable, the second the latent/intelligence one varaible)

[1]Again for every topic the notation changes (λ here has nothing to do with lamdba of PCA)

This factor model can be written in matrix form (considering all the p tests to be explained for a single unit) as

$$\underset{p\times 1}{\mathbf{X}} = \underset{p\times m}{\Lambda}\underset{m\times 1}{f} + \underset{p\times 1}{\mu}$$

where:

- the matrix $\mathbf{\Lambda}$ is the *factor loading matrix* ($p \times m$ because of p observed variables and m latent ones)
- $\mathbf{f}$ is the vecctor of *common factors*
- $\mathbf{u}$ is the vector of *unique factors*

Again we are in a single unit (with p observed variables to be explained) not a sample of units.
The matrix Lambda will be:

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_{11} & \lambda_{12} & \dots & \lambda_{1k} & \dots & \lambda_{1m} \\ \dots \\ \lambda_{i1} & \lambda_{i2} & \dots & \lambda_{ik} & \dots & \lambda_{im} \\ \dots \\ \lambda_{p1} & \lambda_{p2} & \dots & \lambda_{pk} & \dots & \lambda_{pm} \end{bmatrix}$$

Again all what appears on the right hand side of $=$ is unknown: in order to be able to estimate the model we need to put constraints on the parameters/the unknown random variables/vector (f and u).[2]

4.2 Assumptions

We have the following assumptions: the first twos are just symplyfing while the last two are the most important:

1. we assume to work with 0 centered random variables, that is

$$\mathbb{E}\left[\mathbf{f}\right] = \mathbf{0} \quad \mathbb{E}\left[\mathbf{u}\right] = \mathbf{0}$$

   This is a low-cost simplyfing assumption; result can be easily obtained by working on mean centered observations.
   If I had not been working with mean center obs

TODO: check 11

$$X_1 = \mu_i + \lambda_{i1} f_1 + \lambda_{im} f_m$$

   but this complicate the model and adds nothing

$$\underbrace{E(X_1)}_{E(X_1-\mu_1)=0} = \lambda_1 \underbrace{\mathbb{E}\left[f_1\right]} = 0 + \dots \lambda_{im} \underbrace{Ef_m}_{=0} + \underbrace{Eu}_{=0}$$

[2]When we estimate a model, to estimate we always make assumptions: eg in regression epsilon has 0 mean, constant variance and uncorrelated between obs. We need similar conditions here, but on $\mathbf{f}$ and $\mathbf{u}$ (while in regression was only on ε)

2. variance covariance matrix of latent variables is identity

   $$E(ff^T) = V(f) = I_m$$

   The first is the variance covariance matrix of $\mathbf{f}$ since $\mathbf{f}$ is mean centered. This means that we assume that the common factors are assumed to have unit varaince and to be uncorrelated
   This again is a simplyfiny assumptions: its not necessarily but it can be proved (using Choleski decomposition) that any model can be transformed to a model respecting this condition

3. this is a new fondamental condition

   $$\begin{aligned}\mathbb{E}\left[\mathbf{fu}^T\right] &= \underset{m\times p}{\mathbf{0}}\\ \mathbb{E}\left[\mathbf{uf}^T\right] &= \underset{p\times m}{\mathbf{0}}\end{aligned}$$

   these are null matrices of different sizes: actually these are are the variance/covariance between f and u so $\text{Cov}\left(\mathbf{f}, \mathbf{u}\right) = \mathbf{0}$. These tells us that **common factors and the unique factors are uncorrelated**.
   Eg there's no correlation between intelligence and other aspect/noise that impact on performance of a score

4. finally

   $$\mathbb{E}\left[\mathbf{uu}^T\right] = \text{Var}\left[u\right] = \mathbf{\Psi}$$

   with $\mathbf{\Psi}$ a diagonal matrix such as

   $$\mathbf{\Psi} = \begin{bmatrix} \psi_{11} & & 0 \\ 0 & \psi_{ii} & 0 \\ 0 & & \psi_{pp} \end{bmatrix}$$

   This last assumption is that the unique factors are uncorrelated among them (so other things influencing my performance on classics are not correlated with other things influencing my performance in math).
   These variances ψ_{ii} are called **uniquenesses** or **specific variances**: so we have specific/different variances for each factor

TODO: CHECK

Now assuming that all those conditions are met/hold, we're able to compute, as a consequence of the conditions the covariance between the observed variables as

$$\begin{aligned}\mathbb{E}\left[XX^T\right] \overset{(0)}{=} \Sigma &= \mathbb{E}\left[(\mathbf{\Lambda f} + \mathbf{u})(\mathbf{\Lambda f} + \mathbf{u})^T\right]\\ &= \mathbb{E}\left[\mathbf{\Lambda ff}^T\mathbf{\Lambda}^T + \mathbf{\Lambda fu}^T + \mathbf{uf}^T\mathbf{\Lambda}^T + \mathbf{uu}^T\right]\\ &\overset{(1)}{=} \mathbf{\Lambda}\,\mathbb{E}\left[\mathbf{ff}^T\right]\mathbf{\Lambda}^T + \mathbf{\Lambda}\,\mathbb{E}\left[\mathbf{fu}^T\right] + \mathbb{E}\left[\mathbf{uf}^T\right]\mathbf{\Lambda}^T + \mathbb{E}\left[\mathbf{uu}^T\right]\\ &\overset{(2)}{=} \mathbf{\Lambda\Lambda}^T + \mathbf{\Psi}\end{aligned}$$

where in

- (0) the expected value is still covarince matrix becacuse dealing with mean centered data (so Σ)

- (1) we split the expected value; Λ are constant and can be put outside expectation
- (2) we substituted $\mathbb{E}\left[\mathbf{f}f^T\right] = \mathbf{I}$ (for assumption 2), $\mathbb{E}\left[\mathbf{f}\mathbf{u}^t\right] = \mathbf{0}$ and $\mathbb{E}\left[\mathbf{u}\mathbf{f}^t\right] = \mathbf{0}$ (for assumption 3) and $\mathbb{E}\left[\mathbf{u}\mathbf{u}^T\right] = \mathbf{\Psi}$

Thus if the factor model holds, Σ (the covaraince matrix of observed variables)

- is $= \Lambda\Lambda^T + Psi$
- as $\mathbf{\Psi}$ is diagonal, this means that the observed covarainces (off diagonal elements in Σ) are completely determined/accounted for/by the common factors

This means that if a model like the one holds, than we can explain correlation using only factors (since the remaining Ψ is diagonal).

TODO: CHECK

Remark 10. Recappino:

- Factor analysis ($X = \mathbf{\Lambda f} + \mathbf{u}$) is a linear model aimed at explaining *observed* correlation (between observed variables)
- $\mathbf{u}$ has the same role of residual ε in linear models
- as in PCA this model perform dimension reduction: observed variables are p while latent factors are m which is expected to be smaller than p
- specific elements of this model is that all what is on the rhs is unknown/unobserverd (different from linear models where ε is the only unknown/random)
- in order to obtain estimates we need to put constraints on f, u and on their relationship. The first assumption eases the estimation of a simpler model without loss of generality; the second assumption is not fundamental (we will relax it); always possible to transform a model without to a model enforcing this
- if the data are generated under the factor model then we can write Σ as linear combination of Λ and Ψ. Very important that $\mathbf{\Psi}$ is diagonal to say that there's no residual correlation

Given our assumptions on f and u we obtain that if the data have been generated according to model $\mathbf{X} = \mathbf{\Lambda f} + \mathbf{u}$, then $\mathbf{\Sigma} = \mathbf{\Lambda\Lambda}^T + \mathbf{\Psi}$. So to expand a bit

$$\Sigma = \mathbf{\Lambda\Lambda}^T + \mathbf{\Psi}$$

$$\begin{bmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1p} \\ & \sigma_{22} & \dots & \sigma_{2p} \\ & & \sigma_i^2 & \sigma_{ip} \\ & & & \sigma_p^2 \end{bmatrix} = \begin{bmatrix} \lambda_{11} & \dots & \lambda_{1k} & \dots & \lambda_{1n} \\ \dots \\ \lambda_{i1} & \dots & \lambda_{ik} & \dots & \lambda_{in} \\ \dots \\ \lambda_{p1} & \dots & \lambda_{pk} & \dots & \lambda_{pn} \end{bmatrix} \begin{bmatrix} \lambda_{11} & \dots & \lambda_{i1} & \dots & \lambda_{p1} \\ \dots \\ \lambda_{1k} & \dots & \lambda_{ik} & \dots & \lambda_{pk} \\ \dots \\ \lambda_{1n} & \dots & \lambda_{in} & \dots & \lambda_{pn} \end{bmatrix} + \begin{bmatrix} \psi_{11} & 0 & 0 \\ 0 & \psi_{ii} & 0 \\ 0 & 0 & \psi_{pp} \end{bmatrix}$$

Now let's focus on generic element σ_i^2 of Σ it's generated considering the i-th row of $\mathbf{\Lambda}$ and i-th column of $\mathbf{\Lambda}^T$ (and the eleemnt ii of Ψ). So we have

$$\sigma_i^2 = \underbrace{\sum_{k=1}^{m} \lambda_{ik}^2}_{h_i^2} + \psi_{ii} = \lambda_{i1}^2 + \ldots + \lambda_{ik}^2 + \ldots + \lambda_{in}^2 + \psi_{ii}$$

so if the factor model holds, the **observed variance can be decomposed** to part due to the components/lambda variables (common part) and part due to psi (unique part) The quantity:

TODO: cHECK

- $h_i^2 = \sum_{k=1}^m \lambda_{ik}^2$ is called *communality*. the communality is the part of the observed variance that is accounted for by the common factor
- ψ_{ii} the uniqueness: the uniqueness is related to the specific/unique factor

So we can say that the observed variance can be decomposed in the sum of communality + uniqueness.
The latent variables are able to explain just a part of the observed variablity then there's the specific part off course

Focusing on the off diagonal elements, say σ_{ip} this is obtained multiplying the i-th row of $\mathbf{\Lambda}$ and p-th, last column, thus

$$\sigma_{ip} = \sum_{k=1}^{m} \lambda_{ik}\lambda_{pk} + 0$$

(at the end added 0 because off diagonale element of $\mathbf{\Psi}$, $\psi_{ip} = 0$) so here if the model holds, all the observed covariance are fully explained by the observed covariances

TODO: in general CHECK

4.3 Factor loading matrix

Let's focus on $\mathbf{\Lambda}$: we've said that it is the factor loading matrix. What does this mean?
In the first row we have the contribution of each common factor to variable X_1. It tells how much each factor "loads" to obtained the observed variables (look at the model in extended form); it might be that not all the factor are relevant for a certain observed variable.

Now let's derive the expected value of

$$\mathbb{E}\left[\mathbf{X}\mathbf{f}^T\right] = \text{Cov}\left(\mathbf{X}, \mathbf{f}\right)$$

this is the covariance (as alla the variables have zero mena) between the observed variable $\mathbf{X}$ and the common factor $\mathbf{f}$. If $\mathbf{X} = \mathbf{\Lambda}\mathbf{f} + \mathbf{u}$, then the expected value above, to apply the definition

$$\begin{aligned}
\mathbb{E}\left[\mathbf{X}\mathbf{f}^T\right] &= \mathbb{E}\left[(\mathbf{\Lambda}\mathbf{f} + \mathbf{u})\mathbf{f}^t\right] \\
&= \mathbb{E}\left[\mathbf{\Lambda}\mathbf{f}\mathbf{f}^T + \mathbf{u}\mathbf{f}^T\right] \\
&\overset{(1)}{=} \mathbf{\Lambda}\underbrace{\mathbb{E}\left[\mathbf{f}\mathbf{f}^T\right]}_{\mathbf{I}} + \underbrace{\mathbb{E}\left[\mathbf{u}\mathbf{f}^T\right]}_{\mathbf{0}} \\
&= \mathbf{\Lambda}
\end{aligned}$$

where in

- (1) we take out Λ (which acts as constant)
- (2) we substitute from assumptions $\mathbb{E}\left[\mathbf{uf}^T\right] = \mathbf{I}$ and $\mathbb{E}\left[\mathbf{uf}^T\right] = \mathbf{0}$

So beside being the factor loading matrix, Λ is *also the covariance matrix between observed variable X and latent/common factor* $\mathbf{f}$.

4.4 Equivariance

The linear factor model is equivariant with respect to scale changes Assume that

$$\mathbf{Y} = \mathbf{CX}$$

where C is a scale change matrix which is square and diagonal: so we change unit by premultipling our variable by the matrix $\mathbf{C}$ (we multiply each column of X by a constant). It turns out that:

$$\mathbf{Y} = \mathbf{CX} = C(\Lambda_X f + u) = \underbrace{C\Lambda_X}_{\Lambda_Y} f + Cu = \Lambda_Y f + Cu$$

I have a new factor model with the same common factor f, but the lambda matrix has been rescaled. $\mathbf{\Lambda_Y}$ is the original but rescaled applying $\mathbf{C}$ The new factor loading matrix is the old factor loading matrix times the scale change. This proves that linear factor model is scale equivariant. **TODO**: CHECK
If we rescale we don't need to refit the model but just multiply the factor loadings for the matrix of $\mathbf{C}$.

What happens to variance of Y?

$$\begin{aligned}\mathrm{Var}\left[\mathbf{Y}\right] &= \mathbf{C}\,\mathrm{Var}\left[\mathbf{X}\right]\mathbf{C}^T \overset{(1)}{=} \mathbf{C\Sigma C} = C(\Lambda_X\Lambda_X^T + \Psi_X)C \\ &= C\Lambda_X\Lambda_X^T C + C\Psi_X C\end{aligned}$$

where in substitution (1) we removed transposition from $\mathbf{C}$ since it's symmetric. So the same happens with variance: we don't need to reestimate but just take the estimate from X and, if we assume that as transformation we take the standardizing matrix ($1/\sqrt{\sigma}$ you know)

$$C = \Delta^{-1/2}$$

and define

$$Y = \Delta^{-1/2}X$$

Y are standardized variable because have zero mean but also unitary variance/sd

$$\Lambda_Y = \Delta^{-1/2}\Lambda_X$$

This means that the factor loading matrix referred to the standardized variables is equal to the factor loading matrix of the raw data multiplied by $\Delta^{-1/2}$

Interesting: we can model on the unstandardized data and go in the model with standardized variable by just preapplying

TODO: CHECK

So either i premultiply by $\Delta^{-1/2}$ to go to the std data or $\Delta^{1/2}$ to the unstandardized data.
This is not the case with PCA

we can always move from the solution on the original data to the one on the standardized data by just performing a simple scale change (with $\Delta^{-1/2}$)

what is Λ_Y? as

$$\Lambda_Y = \Delta^{-1/2}\Lambda_X = \Delta^{-1/2}\operatorname{Cov}(X,f) = \Delta^{-1/2}\operatorname{Cov}(X,f)\underbrace{I^{-1/2}}_{\text{sd of f}}$$

TODO: CHECK

so Λ_Y is the correlation matrix between X and f, $\Lambda_Y = \operatorname{Corr}(X,f)$

If we work on original data Λ_X is the covariance matrix between the observed and the latent variables; if otherwise we work on the standardized data the Λ_Y is the correlation matrix between the observed and the latent variables.

4.5 Model identifiability

Before moving to estimation issues we need to focus a little bit on what is known as identifiability.

Definition 4.5.1. A model is said to be identifiable if a solution to the estimation problem *exists* and *it is unique*.

Let's focus of a single solution existence: actually our condition are not enough to prove that there is just one solution, there are infinity of models satisfying all the conditions above and a single factor model is not identifiable. So anyone trying to estimate a model obtain a different estimate.
Now we will prove that the solution to the estimation problem is not unique: this is because the linear factor model is invariant after orthogonal rotations. There's an infinity of factor models.
Seems a drawback but it will be useful in the future. Once we fix this problem it turns out to be a benefit.
Let's consider an orthogonal matrix called $\mathbf{G}$ such that

$$\mathbf{G}^T\mathbf{G} = \mathbf{G}\mathbf{G}^T = \mathbf{I}$$

now let's consider our model

$$\begin{aligned} X &= \Lambda f + u \overset{(1)}{=} \Lambda I f + u \\ &\overset{(2)}{=} \underbrace{\Lambda G}_{\Lambda^*}\underbrace{G^T f}_{f^*} + u \end{aligned}$$

where in (1) we inserted an I and in 2 we just replaced by GG^T. Calling ΛG as Λ^* and $G^T f$ as f^* we end up with

$$X = \Lambda^* f^* + u$$

TODO: CHECK

So we can take infinite number of orthogonal matrixes and we obtain two models that has the same properties (f^* has the same property of f, and it means that

Let's prove the two models

$$X = \Lambda f + u$$
$$X = \Lambda^* f^* + u$$

are completely equivalent. In fact start by computing expected valuees

$$\mathbb{E}\left[f^*\right] = \mathbb{E}\left[G^T f\right] = G^T \cdot \underbrace{\mathbb{E}\left[f\right]}_{=\mathbf{0}} = \mathbf{0}$$
$$\mathbb{E}\left[f^* f^{*T}\right] = \mathbb{E}\left[G^T f f^T G\right] = G^T \cdot \underbrace{\mathbb{E}\left[f f^T\right]}_{\mathbf{I}} G = G^T G = I$$

again the common factor have unit variances and are uncorrelated as the ones considered before.

Final check: what's the relationshib between f and u

$$\mathbb{E}\left[f^* u^T\right] = \mathbb{E}\left[G^T f u^T\right] = G^T \underbrace{\mathbb{E}\left[f u^T\right]}_{=\mathbf{0}} = \mathbf{0}$$

so we proved that common factor and unique factor are uncorrelated so teh two models are indistingushiable, have the same property.

Point is that We have an infinity (as the orthogonal matrixes) of equivalent solutions: so people started to try to find an unique solution.
In order to guarantee that a unique solution is obtained we need to fix something else. we impose the constraint that

$$\Lambda^T \Psi \Lambda \quad \text{is diagonal}$$

it is a constraint based on bayesian inference (basically all the software do this): Λ and Ψ cannot be whatever but must be that the quadratic form above is diagonal. By putting this constraint we obtain a single solutions.

When we'll move to interpretation is fine to have a lot of solution.

Existence We proved how to obtain a unique solution, we need to verify if it exists. As seen all what appears on right hand side of $\mathbf{X} = \mathbf{\Lambda f} + \mathbf{u}$ is unknown so there's a lot of things to be estimated.

It's not possible to estimate all in one shot: first we estimate Λ and Ψ (for $\Sigma = \Lambda\Lambda^T + \Psi$) and then we estimate f. Also this problem of estimating Λ and Ψ is a little bit tricky. The starting point of our estimation procedure

$$\Sigma = \Lambda\Lambda^T + \Psi$$

we know the elements of Σ only.
How many are the *unique* elements of Sigma, our pieces of information? It's the sum of first p natural numbers, that is $p(p+1)/2$. We want to reconstruct these elements using $p(p+1)/2$ equations and we want to find the unknows on the other side. Our unknowns are the $p \cdot m$ elements of Λ plus p diagonal elementsof Ψ.
In order to obtain of solutions the number of equations must be larger than the number of unknows;

- on the right hand side we have $p \cdot m + p$ unknowns
- we put a constraint: Λ and Ψ must satisfy $Lambda^T PsiLambda$ is diagonal matrix. This reduces the number of unknown since off diagonal elements fixed to be zero: the matrix is $m \times m$ so we have $m \cdot (m-1)/2$ element fixed/constrainted to 0

$$\frac{p(p+1)}{2} \quad \text{equations}$$
$$pm + p - \frac{m(m-1)}{2} \quad \text{unknowns}$$

In order for a solution to exists

$$\frac{p(p+1)}{2} > pm + p - \frac{m(m-1)}{2} \tag{4.1}$$

p is given (number of variables we have), we need to chose m such that this equality is satisfied (so not all the value of m).
This is the condition of maximum number of common factor we can include in the model: this condition is known in literature as Ledermans condition which puts a limit on the maximum number of common factor that can be included in a factor model with p observed variable.
In order to have solution one could set $\geq$ in the ledermans inequality: the reason we put strict $>$ is that we want to have dimension reduction as well.

TODO: check

4.6 Estimation of the linear factor model

We estimate our population covariance matrix Σ using S (using either the correct or biased version), while ρ is estimated using R
The model is complicated and estimation is performed in two stages:

1. we start the estimation of the model by trying to estimate Lambda and Ψ first we try to estimate this decomposition
$$\Sigma = \Lambda\Lambda^T + \Psi$$
2. then we estimate the factor scores

Starting with the first we try to estimate the decomposition on sample quantities

$$S = \hat{\Lambda}\hat{\Lambda}^T + \hat{\Psi}$$

how to decompose S in the sum above? This estimation issue can be tackled in at least **two different ways**/approaches. We see

1. *principal factor method*: it is a distribution free method, we don't need to make any assumption on the probability density function of $\mathbf{X}$ in the population;

2. *maximum likelihood method*, that assumes normality, $\mathbf{X}$ is normally distributed

Both methods need to start from a preliminary solution and refine it until they find a good solution: so we start from a plausible estimate of the communalities. Remembering the communalities (part of observed variance accounted for by the common factors, parts of variability shared with other variables) are:

$$h_i^2 = \sum_{k=1}^{m} \lambda_{ik}^2$$

that is for each row of Λ we have the sum of its squares.

So communality of X_i is part of the variance of X_i which is explained by the common factors (shared by all the variables). As starting point for it we can adopt different strategies, which are different regarding the matrix we're working with (correlation R of standardized variables or covariance S of unstandardized variables)

1. r-squared-like: to have the part explained we can take the r_{io}^2 of X_i regressed on the *other/remaining* X's. In case of
   - R: we use it directly, that is
   $$\hat{h}_i^2 = r_{io}^2$$
   - in case of S we multiply it by the i-th variable variance to have the quantity explained
   $$\hat{h}_i^2 = s_i^2 \cdot r_{io}^2$$
2. we could use the max (absolute) correlation among X_i and any other variable $X_{i'}$ (absolute value is needed because percentage explained is positive). With
   - R: we use it directly
   $$\hat{h}_i^2 = \max_{i'} |r_{ii'}|$$
   - in case of S we again multiply it by the i-th variable variance
   $$\hat{h}_i^2 = s_i^2 \cdot \max_{i'} |r_{ii'}|$$
3. we just set to the maximum value possible. In this case we get principal component. This means that unique factors does not exists/aren't needed: common factors are able to explain variances/covariances. For
   - for R
   $$\hat{h}_i^2 = 1$$
   - for S:
   $$\hat{h}_i^2 = s_i^2$$

TODO: check

4.7 Principal factor method

Let's start from S. We build what we call the reduced covaraince matrix. At population level we have that

$$\Sigma = \Lambda\Lambda^T + \Psi$$

so i can write

$$\Sigma - \Psi = \Lambda\Lambda^T$$

what happens in the observed case, not population level

$$S = \hat{\Lambda}\hat{\Lambda}^T + \hat{\Psi} \implies S - \hat{\Psi} = \hat{\Lambda}\hat{\Lambda}^T$$

with $S - \hat{\Psi}$ called reduced covariance matrix.
The reduced covariance matrix is identical to S, but have communality on main diagonal instead of variances

$$S - \hat{\Psi} = \begin{bmatrix} \hat{h}_1^2 & s_{12} & \dots & s_{1p} \\ & \hat{h}_2^2 & \dots & s_{2p} \\ \dots & & & \\ & & & h_p^2 \end{bmatrix}$$

Reduced covaraince matrix is still squared symmetric (changed only the diagonal of a symmetric matrix) but no longer positive semidefinite (change diagonal entries). This means that no longer eigenvalue..

TODO: CHECK

As squared symmetric matrix can be decomposed using spectral decomposition

$$S - \hat{\Psi} = \Gamma L \Gamma^T$$

now i consider the positive eigenvalue only. I assume their number is m. So i have that

$$\begin{aligned} S - \hat{\Psi} &\overset{(0)}{\approxeq} \Gamma_m L_m \Gamma_m^T \\ &\overset{(1)}{=} \underbrace{\Gamma_m L_m^{1/2}}_{\Lambda} \underbrace{L_m^{1/2} \Gamma_m}_{\Lambda^T} \end{aligned}$$

where in

- (0) L_m is the diagonal matrix of the positive eigenvalues
- (1) i substituted $L_m = L_m^{1/2} L_m^{1/2}$

So using the spectral theorem I found a way to have an estimate

$$\hat{\Lambda} = \Gamma_m L_m^{1/2}$$

If we want to obtain $\hat{\Psi}$

$$\hat{\Psi} = S - \hat{\Lambda}\hat{\Lambda}^T$$

this matrix

- is know as the residual covariance matrix
- is very important in order to check model fit: if diagonal we're happy, if not our model give poor fit to the data. If large entries are present in the off-diagonal part of $\hat{Psi}$, this means that the model gives a poor fit, as the common factors turn out to be unable to explain the observed covariances. We expected Ψ to be diagonal; if $\hat{Psi}$ elements out of main diagonal are small it's still ok

Poor performance could be dued

- take into account too few factors
- relationship between varaiable is not linear. our model is linear

Once estimated the diagonal elements of $\hat{Lambda}\hat{Lambda}^T$ are the final estimates for the communalities.

Scale equivariance Last comment:

- the factor model is scale equivariant
- however we're estimating factor model using spectral decomposition which is not scale equivariant, estimates obtained by the principal factor method are not scale equivariant. Thus we cannot transform estimate obtained with unstandardized variable to obtain the estimate obtained with standardized variables. As a consequence, model fitted on the raw data or on the standardized ones are different and there is no way to go from one solution to the other

Example 4.7.1 (Esempio cartaceo scores)**.** In the sheet there are some pretty high correlation: pupils with high grades on some subject tends to have high grades in other subjects.
Interpretation of the factor loading matrix: We have two latent variables

1. The first factor is more or less equal important in all the variables: this factor people say is a general value of general intelligence (if high, being positively correlated, one performs well everywhere) while low value of this tend to perform poorly. This is generally considered as general intelligence factor
2. the second factor: large values on this factor has very good score on animal house to block design, but one is very bad in math comprehension. This is math/linguistic vs art/picture tendency.
   Furthermore some variables are not relevant here such as geometric design (coefficient close to 0)

Last column is interesting: once fitted the model we can obtain final estimates for the commualities: that is

$$\begin{aligned}\hat{h}_1^2 &= \lambda_{11}^2 + \lambda_{12}^2 \\ 0.713 &= 0.776^2 + (-0.333)^2\end{aligned}$$

How good is my model in reproducing the observed correlation? in general we have that

$$\sigma_{ii'} = \sum_{k=1}^{n} \lambda_{ik}\lambda_{i'k}$$

we want to study the correlation between information and vocabulary which is $\mathrm{Corr}\,(Information, vocabulary) = 0.755$ (the true correlation in the table) how good is my model to reconstruct the observed 0.755? The reconstructed correlation is

$$\mathrm{Corr}\,(Informa\hat{tion}, Vocabulary) = 0.776 \cdot 0.823 + 0.333 \cdot 0.224 = 0.71324$$

So if i take the difference of the two

$$0.755 - 0.71324 = 0.04176$$

This final one is the entry in the residual correlation of information and vocabulary: given a matrix with all these residual I can study it to know how well my model fit the data.
In the second page of the sheets the matrix are those residual values (not exactly the same because estimated using ML, while we used principal factors method above). All the elemnts are very close to 0; the model gives a pretty good fit to the data.

Important remark 13. last point: consider the definition we've given on the communality

$$h_i^2 = \sum_{k=1}^{m} \lambda_{ik}^2$$

which is the part of variance of X_i which is explained by the m common factor. If we split this sum and focus on the single terms of the sums

$$h_i^2 = \lambda_{i1}^2 + \lambda_{i2}^2 + \ldots + \lambda_{im}^2$$

we have that λ_{i1}^2 is the part of variance of $X_i$which is explained by the first common factor. In this case for information the part is 0.776^2; for the second variable it's 0.823^2.
If I consider the trace of covariance matrix I have the total variance.

$$\mathrm{Tr}\, S = \text{Total variance}$$

while in case of standardized variables the total variance is equal to the number of variables ($\mathrm{Tr}\, R = p$).
If I sum with respect to all the variables

$$\sum_{i=1}^{p} \frac{\lambda_{i1}^2}{p}$$

i obtain the *part of total variance explained by factor 1.* So I have an idea of how a factor explain

Important remark 14 (Recappino)*.* We're interesting in fitting $X = \Lambda f + u$: this is a linear model we're interested in, which is aimed at explaining covariances (or correlations)
However there are too many parameters (all right hand side is unknown).
Under some assumptions we're able to decompose, if the model holds, Σ as

$$Sigma = LambdaLambda^T + Psi$$

where Λ is the factor loading matrix, but also it is the covariance matrix between the observed variables and the common factors (if we deal with std variable will be also the correlation matrix between obs value and common factors).
In order to estimate this decomposition $\Sigma = \Lambda\Lambda^T + \Psi$ we estimate Σ with S, then we obtain a preliminary estimate for the communalities, which are the diagonal entries of Σ

$$sigma_i^2 = h_i^2 + psi_{ii}$$

obtaining the reduced covariance matrix $(S - \hat{\Psi})$.

TODO: checkhere

There are two methods to estimate the model:

1. principal factor methods: based on the spectral decomposition of S - hatPsi (which is the reduced covariance matrix, S with diagonal entries replaced with the preliminary communalities). Thus we obtain factor loading matrix etc etc.
   Then we've seen an example; analysis performed on CORRELATION MATRIX. the first factor is positively correlated with all variable (eg a large value will have higher grades in all the subject, so we can think it as a measure of general intelligence. The second factor is a contrast: negative correlated with first block and positively with last: a large score with this will lgave high grade with last variables and low grade with the first. can be interpred as.
   The second part of table gives info on communalities: initial guesses are R_{io}^2, were interested in the last column that is the final values of communality. If we were working on the raw data we needed variances. In general

   TODO: check

$$\sigma_i^2 = \sum_{k=1}^{m} \lambda_{ik}^2 + \psi_{ii}$$

   but with standardized data we have that $\sigma_i^2 = 1$. otherwise it's s_i^2 so we need to know this in advance $(S' = \hat{\Lambda}\hat{\Lambda}^T + \hat{\Psi})$ Out of diagonal entries we have

$$\sigma_{ii'} = \sum_{k=1}^{m} \lambda_{ik}\lambda_{i'k}$$

   with standardized data $\sigma_{ii'}$ is a correlation.
   Using the example, the correlation between X_1 and X_2 explained by the common factor is

$$\hat{r}_{12} = 0.716 \cdot 0.823 + (-0.333) \cdot (-0.224)$$

we hope that $\hat{r}_{12}$ is near to r_{12} observed (0.755). The only difference here is thatwe obtain the residual correlation matrix (such as that in the higher part of page 493 of the example)

We've said that λ_{11}^2 is the part of the variance of X_1 accounted for by f_1, while λ_{12}^2 is the part of variance of X_1 accounted/explained by f_2 and $\hat{h_i}^2 = \lambda_{11}^2 + \lambda_{12}^2$ is the part of the variance of X_1 which is explained by the common factors.
As we're dealing with std vars in this example, the variance of $X_1 = 1$
Now $\hat{lambda}_{21}^2$ is the part of the variance of X_2 accounted for by factor 1 and again the variance of X_2 is 1.
now $\hat{\lambda}_{p1}^2$ is the part of the variance of X_p accounted for by factor 1 and again the variance of X_p is 1.
The trace $\operatorname{Tr} R = p$ is the total variance (number of variables) (if we were dealing with non std variables the total variance would be tr(S)).
if i take the sum

$$\frac{\sum_{i=1}^{p} \hat{\lambda_{i1}^2}}{p} \cdot 100$$

this is the percentage of the total variance which is explained by the first factor f_1. In the sheets that percentage is 48%: the first factor is able to explain the 48% of total valriability.

4.8 ML method

Second method we see for estimation is Maximum Likelihood. When Spearman invented this model, the only estimation available was the previous one (ML was not available, while spectral decomposition was). Then people started saying; i would prefer to have a way to formally test if say two factor are enough; when ML was introduced people reset the problem above with ML.

We need to specify the distribution in the population: tipically we assume that X is MVN with parameters μ, Σ

$$X \sim \text{MVN}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$$

the pdf of MVS is

$$f(\mathbf{x}) = \det 2\pi\boldsymbol{\Sigma}^{-1/2} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

within the exponential we have the mahalanobis distance between a unit and the mean of the distribution.
the likelihood will be (assuming independence of observations

$$L(x; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{n} f(x) = \det 2\pi\Sigma^{-n/2} \exp\left\{-\frac{1}{2}\sum_{j=1}^{n}(\mathbf{x}_j-\mu)^{\mathbf{T}}\boldsymbol{\Sigma}^{-\mathbf{1}}(\mathbf{x_j}-\mu)\right\}$$

the logarithm allows us to get rid of exponential part

$$l(x; \mu, \Sigma) = -\frac{n}{2}\log\det 2\pi\Sigma - \frac{1}{2}\left\{\sum_{j=1}^{n}(\mathbf{x}_j-\mu)^{\mathbf{T}}\boldsymbol{\Sigma}^{-\mathbf{1}}(\mathbf{x_j}-\mu)\right\}$$

what happens is that we can rewrite this quantity as

$$l(x;\mu\Sigma) = -\frac{n}{2}\log\det 2\pi\Sigma - \frac{1}{2}\operatorname{Tr}\Sigma^{-1}S - \frac{n}{2}(\overline{x}-\mu)^T\Sigma^{-1}(\overline{x}-\mu)$$

The ML estimate for mean $\boldsymbol{\mu}$ is sample mean $\bar{\mathbf{x}}$ if we substitute $\overline{x}$ to μ then the quadratics in the third term disappears and we have that

TODO: check non chiarissimo nei miei appunti se ho copiato bene

$$l(x;\mu\Sigma) = -\frac{n}{2}\log\det 2\pi\Sigma - \frac{n}{2}\operatorname{Tr}\Sigma^{-1}S$$

If $\Sigma = \Lambda\Lambda^T + \Psi$ holds, then

$$l(x;\Sigma) = l(x;\Lambda,\Psi) = -\frac{n}{2}\log\det 2\pi(\Lambda\Lambda^T+\Psi) - \frac{n}{2}\operatorname{Tr}(\Lambda\Lambda^T+\Psi)^{-1}S$$

Deriving with respect to Λ and Ψ and set the derivatives $= 0$, unfortunately this equation dont admit a close formed solution: there's no formula to compute Λ and Ψ directly. It took up to the 70's: Joreskoe provided a numeric algorithm to solve the problem, so numeric methods are used to obtain numeeric estimates for Λ and Ψ.

Example 4.8.1. back to the sheet example the estimates using ML are close but not the same

Scale equivariance

One important aspecct is that ML estimates are *scale invariant*: if we fit the factor model with ML on std data, we can go back to factor model on original variables by applying the same scale change.

Differently from previous method (spectral decomposition) both *method* and *estimates* are scale equivariant.
maximum likelihood yields scale equivariant estimates: this means that after fitting the model to the raw data we can obtain the ones referred to the standardized data by simply rescaling them.
Lambda is the covariance matrix the observed variables and the common factors. Let call it $\hat{\Lambda}_X$. If i am interested in $\hat{Lambda}_Z$ (referred to the standardized variables) i simply need

TODO: check

$$\hat{\Lambda}_Z = D^{-1/2}\hat{\Lambda}_X I_m^{-1/2} = D^{-1/2}\hat{\Lambda}_X$$

Λ is a covariance, to obtain a correlation I have to divide by std deviation (at left we have std of X, at right the std of common factor which is standardized).

This transformation can be done only with ML estimates because scale equivariant.

Hypothesis testing

The fact that we're using ML allows us to perform hypothesis testing on number of factors to take: before the decision was based on residual covariance matrix. Wow under normality assumtpion we can test if a given number of factors is enough. we test sequentially: we start with a model with 1 factor. the test is

$$H_0 : sigma...$$
$$H_1 : \text{is unstructured (different from model in } H_0)$$

the test statistics is the W statistics in the example (this is the LRT statistic). As all LRT is distributed asympotetically as Chisq with df which depend on n of variable and factors (p is fixed, m is change via via).
We start testing the hypothesis with 1 factor: is 1 factor enough? we estimate all the components in W if we refuse the null we need to increment by 1 the number of factors

1. At first step we have 0.007: we decide to reject null (that 1 factor is enough) and need to increment number of factors
2. we fit a two factor model: p-value is 0.92 so we don't reject the null and the model is ok to take two factor

TODO: CHECK ultima considerazione

we're seeking linear factor: if we reject the model with maximum number of factor there's some problem with the model, not the number of factors

4.9 Factor rotation

H in the sheet is an orthogonal matrix

if we compute $Lam\hat{b}daH$ (orthogonal rotation) we obtain a new solution in Γ^*

the new model has the same properties of the original one

It was G in the notes the orthogonal matrix wi

One way to check that the solution is the same is to compute the communalities: if one do the computations, we obtain the same communality for the first variable.

Same holds for the explained covariances.

this is what invariance after orthogonalrotation means.

Why are we interested in rotations? Thurstone (psychometrician) in 1947 wrote a paper which defined what a *simple factor structure* is.

Definition 4.9.1. If the factor strucuture is simple, the variables should be divisible into gorups such that hte loadings, within each group, are high on a single factor, perhaps moderate to low on a few factors and negligible on the remaining factor.

Eg if we have three common factor with these loadings (H = high, M = moderate, 0 = null)

$$\Lambda = \begin{bmatrix} H & M & 0 \\ H & M & 0 \\ H & M & 0 \\ 0 & H & M \\ 0 & H & M \\ M & O & H \\ M & O & H \end{bmatrix}$$

Interpreting here is easier. this is tipically not the case with unrotated factors where the first is a general (and other are contrasts).

There are a lot of rotations: most widely known are

- varimax (orthogonal, uncorrelated factors)
- quartimax (orthogonal, uncorrelated factors)
- oblimin (not orthogonal, correlated)

The first twos maximizes something, the last minimize something. In the last obli means *oblique* rotations so it provides correlated factors)

4.9.1 Variamax

An idea of how VARIMAX works. We maximize a function V

$$V = \sum_{k=1}^{m} \left[\overbrace{\underbrace{\frac{\sum_{i=1}^{p} \beta_{ik}^4}{p}}_{(A)} - \underbrace{\left(\frac{\sum_{i=1}^{p} \beta_{ik}^2}{p} \right)^2}_{B}}^{C} \right]$$

where

$$\beta_{ik} = \frac{\lambda}{\sqrt{\sum_{k=1}^{m} \lambda_{ik}^2}} = \frac{\lambda_{ik}}{h_i}$$

Starting from β_{ik}: numerator is usual loading of variable i on factor k. Denominator is sum of squared loadings in the row with sqrt (sqrt of communality), it's the norm of the row vector

So the ratio is a kind of normalization of λ_{ik} by the norm so the matrix which is obtained in this way has row that has unit norm . **TODO**: CHECK

Now going to the main equation:

- summation works inside the columns of the obtained normalized matrix: within parenthesis there's a sort of variance
- both each components inside square brackets are shit column wise: at the numerator we have the fast variance formula, inside squared brackets i have variance of squared beta values in column k

so in essence looking at the main equation of V:

- (A) is the square of quadratic mean of β
- (B) is the arithmetic mean of squared β (kind of variance)
- C is variance of β values in column k.

Point is: **we maximize the sum of the variances inside of each column**.

We obtaine large weights on some factor and low weights on other one.

If the original solution obtained is difficult to interpret, we can apply this transformation to ease/optimize it.

4.10 Factor score estimation

Remembering our factor model was

$$X = \Lambda f + u$$

now we are able to estimate Λ by $\hat{\Lambda}$ (using ML or the other).
We **want now to estimate** f, in order to end with an estimate of $\mathbf{u}$.

I can say that for factor k, the score for factor k is obtainable as linear combination of value of $\mathbf{x}$: i can measure how good is f_k^* in predicting f_k.

4.10.1 Thompson

I want to find f_k^* such as :

$$\mathbb{E}\left[(f_k^* - f_k)^2\right] \quad \text{is the smallest}$$

This quanity is a funciton of a_k

$$\phi = \mathbb{E}\left[(a_k^T x - f_k)^2\right]$$

we want to find a_k for which ϕ is minimum; so go with derivatives and set $= 0$

$$\frac{\partial \phi}{\partial a_k} = \mathbb{E}\left[2x(x^T a_k - f_k)\right] = 2\,\mathbb{E}\left[xx^T\right] a_k - 2\,\mathbb{E}\left[x f_x\right]$$

where $\mathbb{E}\left[xx^T\right]$ is cov matrix of X and with mean centered data $\mathbb{E}\left[x f_x\right]$ is variance/covaraince between X and f_x thus the k-th column of Λ (call it Λ_k) so we have

$$\frac{\partial \phi}{\partial a_k} = 2\Sigma a_k - 2\Lambda_k = 2(\Sigma a_k - \Lambda_k)$$

by putting $2(\Sigma a_k - \Lambda_k) = 0$ and solving for a_k we obtain

$$a_k = \Sigma^{-1}\Lambda_k$$

So

$$f_k^* = a_k^T x = \Lambda_k^T \Sigma^{-1} x$$

then making this for all the factor $k = 1, \ldots, m$ and collect in matrix i have

$$F^* = \Lambda^T \Sigma^{-1} x$$

it can be proved that F^* can equivalently be written as

$$F^* = \left(\mathbf{I} + \Lambda^T \Psi^{-1} \Lambda\right)^{-1} \Lambda^T \Psi^{-1} X$$

this formula is known as *Thompson estimator* (estimator for factor scores).

Properties of estimator What are the property of this estimator?

- minimize the squared prediction error (by definition/derivation): it has been obtained by minimizing the expected squared prediction error. But if we compute the expected value

$$\begin{aligned}\mathbb{E}\left[F^*|F\right] &= \mathbb{E}\left[(\mathbf{I}+\Lambda^T\Psi^{-1}\Lambda)^{-1}\Lambda^T\Psi^{-1}X|F\right] \\ &= \underbrace{(I+\Lambda^T\Psi^{-1}\Lambda)^{-1}\Lambda^T\Psi^{-1}}_{\text{all constants}}\cdot\mathbb{E}\left[X|F\right]\end{aligned}$$

  but considering $X = \Lambda f + u$

$$\mathbb{E}\left[X|F\right] = \mathbb{E}\left[\Lambda f + u\right] = \Lambda f + \underbrace{\mathbb{E}\left[u\right]}_{=0} = \Lambda f$$

  with Λf constant since conditioning. So coming back we have that

$$\mathbb{E}\left[F^*|F\right] = \underbrace{(I+\Lambda^T\Psi^{-1}\Lambda)^{-1}\Lambda^T\Psi^{-1}\Lambda}_{\neq I} F \neq F$$

  So $EF^*|F \neq F$ and this estimator is biased

TODO: buono in termini di prediction error? controllare registrazioni

4.10.2 Bartlett estimator

Another solution for estimating F resorting to Bartlett's estimator. Our model is still $X = \Lambda f + u$ (here f a vector for a single variable) and $\text{Var}\left[u\right] = \Psi$.
If estimate Λ using $\hat{\Lambda}$ this looks like a linear regression model with uncorrelated but heteroscedastic (different variance) error; in regression residuals were expected to be omoschedastic ($\text{Var}\left[epsilon\right] = \sigma^2 I$).
Here they're not homoscedastic: with non omoschedasticity we can estimate it use *weighted least square* (weight: residual with large variance have lower weight) the function we want to minimize is

$$u^T\Psi^{-1}u = (x-\Lambda f)^T\Psi^{-1}(X-\Lambda f)$$

derive with respect fo f and set the derivatives = 0

$$\frac{\partial}{\partial f} = -2\Lambda^T\Psi^{-1}(X-\Lambda f) = -2\Lambda^T\Psi^{-1}X + 2\Lambda^T\Psi^{-1}\Lambda f$$

Setting the last = 0 we have

$$\Lambda^T\Psi^{-1}\Lambda f = \Lambda^T\Psi^{-1}X$$

in order to obtain f we need to calculate the inverse of $\Lambda^T\Psi^{-1}\Lambda$. So the estimator of bartlett is

$$\hat{f} = (\Lambda^T\Psi^{-1}\Lambda)^{-1}\Lambda^T\Psi^{-1}X$$

Properties we have that

$$\begin{aligned}\mathbb{E}\left[\hat{f}|f\right] &= \mathbb{E}\left[(\Lambda^T\Psi^{-1}\Lambda)^{-1}\Lambda^T\Psi^{-1}X|f\right] \\ &= (\Lambda^T\Psi^{-1}\Lambda)^{-1}\Lambda^T\Psi^{-1}\cdot\underbrace{\mathbb{E}\left[X|f\right]}_{\Lambda f} \\ &= (\Lambda^T\Psi^{-1}\Lambda)^{-1}\Lambda^T\Psi^{-1}Lambdaf \\ &= If = f\end{aligned}$$

so we have that it is unbiased estimator, but compared to the previous has larger varaince: previous has the lowest variance but has bias.

4.11 Relationship between PCA and Factor Analysis

One option we have when setting preliminary guess of communality is to set = 1 (with standardized data) or observed variance (if working with raw data).

the Principal factor method worked on reduced covariance matrix

$$S - \hat{\Psi} = \begin{bmatrix} h_1^2 & s_{12} & \dots & \dots \\ & h_2^2 & \dots & \\ & & & h_p^2 \end{bmatrix}$$

If we decide that preliminary communalities (on the diagonal) are the observed variance (in diagonal) the reduced covariance matrix is just S, just the variance covariance matrix.
The same happens if we set it to 1 when working with std data, we obtain the original correlation matrix

so in this cases the reduced covariance correlation matrix is equal to the original covariance/correlation matrix)

Principal factor method perform spectral decomposition of the reduced covariance matrix, now since this coincides with original covariance correlation it's just a PCA.

As according to our choice for the communialities $S - \hat{\Psi}$ and S coincide, extracting the factor using principal factor method amounts to perform the spectral decomposition of S, that is to obtain PCs

So setting communality = 1and principal factor method we obtain PCA.

BUT assume we have X which is $p \times 1$ and the principal components $y = A^T x$ with A has eigenvector in column.
Since A is orthogonal it inverted coincides with transpose, by premultipluing A^T with its invert

$$X = Ay$$

i split matrix **A** into two parts

$$A = \begin{bmatrix}\mathbf{A}_m & \mathbf{A}_{p-m}\end{bmatrix}$$

with A_m the first m eigenvector and $\mathbf{A}_{p-m}$ the remaining ones. Y can be splitted vertically

$$Y = \begin{bmatrix} Y_m \\ Y_{p-m} \end{bmatrix}$$

this means that X can be written as

$$X = Ay = \begin{bmatrix} \mathbf{A}_m & \mathbf{A}_{p-m} \end{bmatrix} \begin{bmatrix} Y_m \\ Y_{p-m} \end{bmatrix} = A_m Y_m + A_{p-m} Y_{p-m}$$

i can also put I in the right place and decompose it as $L_m^{1/2} L_m^{-1/2}$ (with L_m a diagonal matrix with the first m eigenvalue)

$$\begin{aligned} X &= A_m I_m Y_m + A_{p-m} Y_{p-m} \\ &= A_m L_m^{1/2} L_m^{-1/2} Y_m + A_{p-m} Y_{p-m} \end{aligned}$$

In so doing I separated first m principal compoenents on one side and the remaining on the other size. Now let's call

$$\begin{aligned} &A_m L_m^{1/2} = \Lambda \\ &L_m^{-1/2} Y_m = f \\ &A_{p-m} Y_{p-m} = \eta \end{aligned}$$

So we have that the above can be rewritten as

$$X = \Lambda f + \eta$$

Is this a linear factor model? if the answer is yes PCA is a special case of factor analysis, otherwise they are two different things. The answer is that **this is not a linear factor model**.
To check it we check whether latent variables involved in the model satisfy all the assumption; especially we want to check the linear assumptions we made

1. is true that $\mathbb{E}\left[(\right] f) = \mathbf{0}$? we have that

$$\mathbb{E}\left[L_m^{-1/2} Y_m\right] = \mathbf{0}$$

because we're working with mean centered data. So this first condition is satisfied

2. is common factor uncorrelated and have unit variance, that is $\mathbb{E}\left[ff^T\right] = \mathbf{I}$? we have

$$\mathbb{E}\left[ff^T\right] = \mathbb{E}\left[L_m^{-1/2} Y_m Y_m^T L_m^{-1/2}\right] \overset{(=)}{1} L_m^{-1/2}\, \mathbb{E}\left[Y_m Y_m^T\right] L_m^{-1/2}$$

whhere in (1) we take $L_m^{-1/2}$ out of expectation. WHat is $\mathbb{E}\left[Y_m Y_m^T\right]$? It's the variance/covariance of the first m principal components, a diagonal matrix with eigenvalues on diagonal, L_m. So we have

$$\mathbb{E}\left[ff^T\right] = L_m^{-1/2} L_m L_m^{-1/2} = L_m^{-1/2} L_m^{1/2} L_m^{1/2} L_m^{-1/2} = \mathbf{II} = \mathbf{I}$$

so ok, the property is verified

3. is $\mathbb{E}\left[eta\right] = \mathbf{0}$? We have

$$\mathbb{E}\left[\eta\right] = \mathbb{E}\left[A_{p-m} Y_{p-m}\right] = A_{p-m}\,\mathbb{E}\left[Y_{p-m}\right] \overset{(1)}{=} \mathbf{0}$$

where in (1) we have that $\mathbb{E}\left[Y_{p-m}\right] = \mathbf{0}$ because mean centered.
So even this this is satisfied

4. property foundamental which relates common factor and unique factors, that is $\mathbb{E}\left[f^T eta\right] = \mathbf{0}$

TODO: non chiarissima la trasposizione qui check

$$\mathbb{E}\left[f^T \eta\right] = \mathbb{E}\left[f \eta^T\right] = \mathbb{E}\left[L_m^{-1/2} Y_m Y_{p-m}^T A_{p-m}^T\right] = L_m^{-1/2}\,\mathbb{E}\left[Y_m Y_{p-m}^T\right] A_{p-m}^T$$

where $\mathbb{E}\left[Y_m Y_{p-m}^T\right]$is the covariance between the first m components and the last p-m ones. This is $\mathbf{0}$ because pcs are uncorrelated so actually all is 0 and check is verified

5. is $\mathbb{E}\left[\eta\eta^T\right]$ a diagonal matrix? remembering

$$\Psi = \begin{bmatrix} \psi_{11} & 0 & 0 \\ 0 & \psi_{ii} & 0 \\ 0 & & \psi_{pp} \end{bmatrix}$$

is $\mathbb{E}\left[\eta\eta^T\right]$ similar to this one? we have

$$\mathbb{E}\left[\eta\eta^T\right] = \mathbb{E}\left[A_{p-m} Y_{p-m} Y_{p-m}^T A_{p-m}^T\right] = A_{p-m}\,\mathbb{E}\left[Y_{p-m} Y_{p-m}^T\right] A_{p-m}^T$$

where $\mathbb{E}\left[Y_{p-m} Y_{p-m}^T\right]$is the variance covariamce of last $p-m$ components, it's a diagonal matrix with last $p-m$ eigenvalues (it's L_{p-m}). It is a diagonal matrix but if the entries/diagonal elements are different this is not enough: in general diagonal elements are different so the whole product is not diagonal, that is $\mathbb{E}\left[\eta\eta^T\right]$ is not diagonal.
The condition does not hold: the unique factor η are not uncorrelated, and this means that in the model

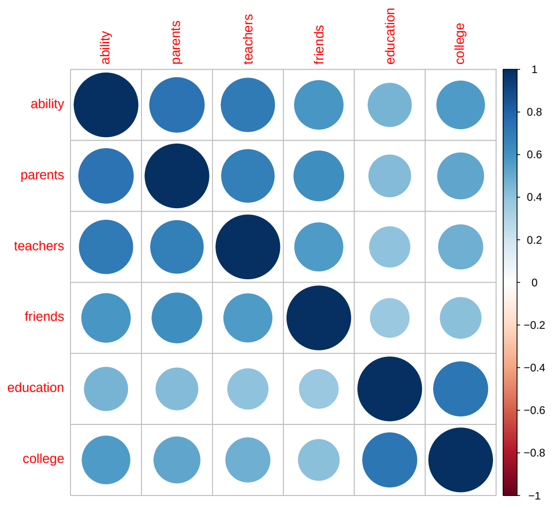
$$X = \Lambda f + \eta$$

the commond factors f are not able to explain the observed covariances (which was the goal of fitting a factor model)

Important remark 15. If we decide to put communality $= 1$ or observed variances the algo fits principal components, but this is not a model: its just PCA because it does not satisfy all the requirements of FA.
People wrongly think that what can be done in FA can be done in PCA (eg rotating); this is just wrong. the property of Linear factor model does not holds with PCA

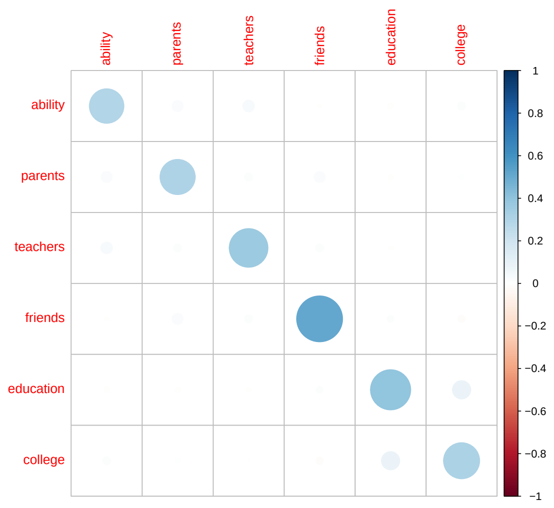
4.12 Lab

```
ability <- read.table("data/ability-education.txt")
## in this case this is a correlation matrix between variables

## need to transform it in a matrix
(R <- as.matrix(ability))

##           ability parents teachers friends education college
## ability      1.00    0.73     0.70    0.58      0.46    0.56
## parents      0.73    1.00     0.68    0.61      0.43    0.52
## teachers     0.70    0.68     1.00    0.57      0.40    0.48
## friends      0.58    0.61     0.57    1.00      0.37    0.41
## education    0.46    0.43     0.40    0.37      1.00    0.72
## college      0.56    0.52     0.48    0.41      0.72    1.00

# we have quite high correlation all positive, it makes sense to perform factor
# analyusis
## with higher number of variable visualizzation plot is useful
library(corrplot)

## corrplot 0.95 loaded

corrplot(R)
```



```
## 1) principal factor method by hand
## first thing to do is to have a preliminary value of communality: we use
## the multiple correlation coefficient 1-1/r_ii* with i-th diagonal element of
## R^_{-1}

(h2tilde <- 1 - 1/diag(solve(R))) # diagonal of the inverse of correlation matrix

##   ability   parents  teachers   friends education   college
## 0.6427569 0.6248924 0.5695938 0.4358076 0.5265227 0.5928205
## at denominator

# these are the preliminary estimates of communalities
## we compute the reduced correlation matrix

redR <- R # replace diag with preliminary estimates of comm
diag(redR) <- h2tilde
redR # no more 1 on diagonal

##              ability   parents  teachers   friends education   college
## ability   0.6427569 0.7300000 0.7000000 0.5800000 0.4600000 0.5600000
## parents   0.7300000 0.6248924 0.6800000 0.6100000 0.4300000 0.5200000
## teachers  0.7000000 0.6800000 0.5695938 0.5700000 0.4000000 0.4800000
## friends   0.5800000 0.6100000 0.5700000 0.4358076 0.3700000 0.4100000
## education 0.4600000 0.4300000 0.4000000 0.3700000 0.5265227 0.7200000
## college   0.5600000 0.5200000 0.4800000 0.4100000 0.7200000 0.5928205

## to compute factor loadings: we need to compute low rank approximation (check)

## to do SVD
spectral <- eigen(redR)
spectral$values # eigenvalues: only first two are positive. this because

## [1]  3.33126929  0.47350875 -0.04368765 -0.08468263 -0.11304912 -0.17096486

# reduced is not positive definite. looking at eigen we can think considering
# that two factor should be enough, m should be 2

## we build the matrix of eigenvector with just 2 eigenvectors and vectors of
## eigenvalue with just 2 eigenvalues,

## then we construnt
## lambda = gamma_m L_m^{1/2} # looking at the used

G <- spectral$vectors[, 1:2]     # gamma only hte first two columns/eigvenvector
L <- diag(spectral$values[1:2]) # diag matrix with first two eigenvalues

G %*% L %*% t(G)  # low rank approximation of correlation matrix

##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 0.7072292 0.7001694 0.6668653 0.5834546 0.4685601 0.5441205
## [2,] 0.7001694 0.6955646 0.6634608 0.5801711 0.4374113 0.5137057
## [3,] 0.6668653 0.6634608 0.6332413 0.5536204 0.4057116 0.4789896
## [4,] 0.5834546 0.5801711 0.5536204 0.4840494 0.3583494 0.4222716
## [5,] 0.4685601 0.4374113 0.4057116 0.3583494 0.6043086 0.6378458
## [6,] 0.5441205 0.5137057 0.4789896 0.4222716 0.6378458 0.6803849

Lambda <- G %*% L^0.5 # factor loadings: G_m L^{1/2}
# a matrix of dimension 6x2 is what we want 2 factor 2 column, 6 variables in
# row. interpret later
```

```
## Now we compute the new communality: communality is part of variance
## explained by common factor. To extract communality,
## Lambda t(Lambda) and extract diagonal (interested only in the varainces not covariances)

(communalities <- diag(Lambda %*% t(Lambda)))

## [1] 0.7072292 0.6955646 0.6332413 0.4840494 0.6043086 0.6803849

## each values is the proportion of variability explained by the two factors
## eg 70% of ability is explained from the two factor

## extract uniqueness: diag of correlation matrix - communality
uniqueness <- diag(R)- communalities # == 1 - communalities

## CHECK THE GOODNESS OF FIT: compute residual correlation matrix (diff between
## observed and Lambda t(Lambda))
R - Lambda %*% t(Lambda) # out of main diagonal should be close to zero

##                ability      parents     teachers      friends    education
## ability    0.292770802  0.029830608  0.033134677 -0.003454555 -0.008560141
## parents    0.029830608  0.304435440  0.016539177  0.029828913 -0.007411273
## teachers   0.033134677  0.016539177  0.366758716  0.016379562 -0.005711590
## friends   -0.003454555  0.029828913  0.016379562  0.515950562  0.011650634
## education -0.008560141 -0.007411273 -0.005711590  0.011650634  0.395691356
## college    0.015879548  0.006294259  0.001010408 -0.012271579  0.082154195
##                college
## ability    0.015879548
## parents    0.006294259
## teachers   0.001010408
## friends   -0.012271579
## education  0.082154195
## college    0.319615088

## test mio
corrplot(R - Lambda %*% t(Lambda))
```



```
## out of diagonal low values

## TOTAL variability explained by each factor, total variability is p. To do
## that is the tr(Lambda) Lambda (Lambda il 6x2 but we want a 2x2: consider the
## transpose of Lamnda)

diag(t(Lambda) %*% Lambda) # interested in diagonal, the variance

## [1] 3.3312693 0.4735087

## so the first factor explain 3.33 and second 0.47

## percentage explained
diag(t(Lambda) %*% Lambda) / 6 # number of variables

## [1] 0.55521155 0.07891812

## first factor explain 55% of total variability, 7% for the second


## finally, to compute communality an equivalent way
apply(Lambda, 1, function(x) sum(x^2)) # same result as before

## [1] 0.7072292 0.6955646 0.6332413 0.4840494 0.6043086 0.6803849
```

```
# how to do in R. fa and factanal functions

library(psych) # psych::fa  gives both principal factor method and ML

##
## Caricamento pacchetto:  'psych'
## Il seguente oggetto è mascherato da 'package:car':
##
##     logit
## Il seguente oggetto è mascherato da 'package:lbmisc':
##
##     table2df

## vohabolario vernaholo giorentino

# n.obs is needed for p-values, we're not using it
# rotate, gives rotation to be considered
# fm: method used for estimation. default seen minres has not be seen in class,
## "pa" principal factor,
## "ml" for maximum likelihoo

pafa1 <- fa(R,
            nfactors = 2, # nfactors wanted (we already seen two positive eigenv)
            rotate = "none", # for the moment we dont' rotatae
            fm = "pa" # principal factor solution as done before
            )

pafa1

## Factor Analysis using method =  pa
## Call: fa(r = R, nfactors = 2, rotate = "none", fm = "pa")
## Standardized loadings (pattern matrix) based upon correlation matrix
##            PA1   PA2   h2   u2 com
## ability   0.83 -0.18 0.73 0.27 1.1
## parents   0.82 -0.24 0.73 0.27 1.2
## teachers  0.77 -0.24 0.66 0.34 1.2
## friends   0.67 -0.19 0.49 0.51 1.2
## education 0.67  0.48 0.68 0.32 1.8
## college   0.76  0.45 0.77 0.23 1.6
##
##                        PA1  PA2
## SS loadings           3.44 0.61
## Proportion Var        0.57 0.10
## Cumulative Var        0.57 0.68
## Proportion Explained  0.85 0.15
## Cumulative Proportion 0.85 1.00
##
## Mean item complexity =  1.3
## Test of the hypothesis that 2 factors are sufficient.
##
## df null model =  15  with the objective function =  3.3
## df of  the model are 4  and the objective function was  0.01
##
## The root mean square of the residuals (RMSR) is  0.01
## The df corrected root mean square of the residuals is  0.02
##
```

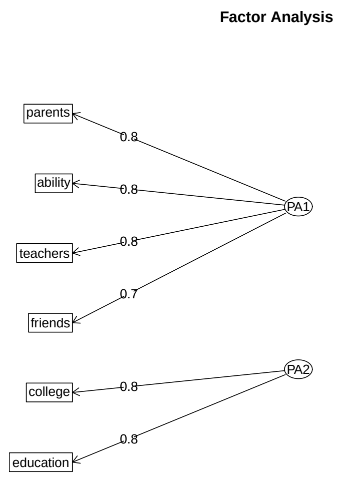
```
## Fit based upon off diagonal values = 1
## Measures of factor score adequacy
##                                                    PA1  PA2
## Correlation of (regression) scores with factors   0.96 0.83
## Multiple R square of scores with factors          0.92 0.68
## Minimum correlation of possible factor scores     0.84 0.36

pafa1$loadings # matrix of loadings

##
## Loadings:
##           PA1    PA2
## ability    0.834 -0.181
## parents    0.822 -0.236
## teachers   0.775 -0.239
## friends    0.672 -0.192
## education  0.666  0.483
## college    0.757  0.445
##
##                  PA1   PA2
## SS loadings    3.440 0.614
## Proportion Var 0.573 0.102
## Cumulative Var 0.573 0.676

Lambda # similar a part of the sign to what obtained before

##            [,1]       [,2]
## [1,] -0.8273744 -0.1506015
## [2,] -0.8103705 -0.1971401
## [3,] -0.7682259 -0.2075338
## [4,] -0.6732548 -0.1754348
## [5,] -0.6452398  0.4335599
## [6,] -0.7281779  0.3874814

## SS loadings are total variability explained
## second row is the proportion of variability

## to extract thecommunalities and uniquenesses
pafa1$communality # similar to what done before

##   ability   parents  teachers   friends education   college
## 0.7292947 0.7305715 0.6570953 0.4882353 0.6776897 0.7714643

pafa1$uniquenesses

##   ability   parents  teachers   friends education   college
## 0.2707053 0.2694285 0.3429047 0.5117647 0.3223103 0.2285357

# residual correlation matrix
pafa1$residual # off diagonal still values very close to 0 (good fit)
```

```
##                ability      parents      teachers      friends     education
## ability    0.270705253  0.001639509  0.0102526111 -0.015507856 -0.0084315471
## parents    0.001639509  0.269428492 -0.0127285853  0.012765292 -0.0034115950
## teachers   0.010252611 -0.012728585  0.3429047263  0.003724221 -0.0007200665
## friends   -0.015507856  0.012765292  0.0037242208  0.511764685  0.0147570912
## education -0.008431547 -0.003411595 -0.0007200665  0.014757091  0.3223103254
## college    0.008871656  0.002985356 -0.0001766904 -0.013570761  0.0002745308
##                 college
## ability    0.0088716563
## parents    0.0029853559
## teachers  -0.0001766904
## friends   -0.0135707613
## education  0.0002745308
## college    0.2285356519
```

```
## since loadings are not rotatad this can be difficult to be interpret, if we
## rotate it helps

pafa2 <- fa(R,
            nfactors = 2,
            rotate = "varimax", # unique change
            fm = "pa"
            )



pafa2
```

```
## Factor Analysis using method =  pa
## Call: fa(r = R, nfactors = 2, rotate = "varimax", fm = "pa")
## Standardized loadings (pattern matrix) based upon correlation matrix
##            PA1  PA2   h2   u2 com
## ability   0.79 0.33 0.73 0.27 1.3
## parents   0.81 0.28 0.73 0.27 1.2
## teachers  0.77 0.25 0.66 0.34 1.2
## friends   0.66 0.23 0.49 0.51 1.2
## education 0.27 0.78 0.68 0.32 1.2
## college   0.36 0.80 0.77 0.23 1.4
##
##                        PA1  PA2
## SS loadings           2.50 1.56
## Proportion Var        0.42 0.26
## Cumulative Var        0.42 0.68
## Proportion Explained  0.62 0.38
## Cumulative Proportion 0.62 1.00
##
## Mean item complexity =  1.3
## Test of the hypothesis that 2 factors are sufficient.
##
## df null model =  15  with the objective function =  3.3
## df of  the model are 4  and the objective function was  0.01
```

```
##
## The root mean square of the residuals (RMSR) is  0.01
## The df corrected root mean square of the residuals is  0.02
##
## Fit based upon off diagonal values = 1
## Measures of factor score adequacy
##                                                    PA1  PA2
## Correlation of (regression) scores with factors   0.91 0.88
## Multiple R square of scores with factors          0.83 0.77
## Minimum correlation of possible factor scores     0.66 0.54

## looking at the loadings, the first factor has high entries for first four
## variables, while the second has high loading fo the last variable
## now this is a dataset on 556 students which answered question:
## - ability: self concept ability
## - parents: perceived parental evaluation
## teachers: perceived teacher evaluation
## friend: perceived friends
## education
## college

# so high entries on first factor which is called perceived ability
# second factor is the attitude of the student towarrd education

# visualization of factor
fa.diagram(pafa2)
```



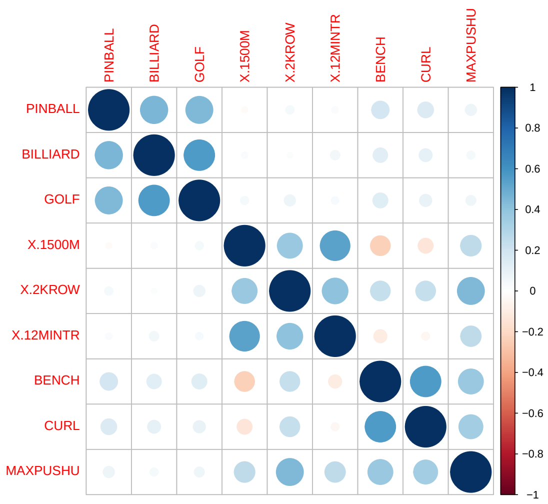
```
## PA1 first factor is highly related to the first four variables while second
## to the last two
```

```
# let's consider ML method

mlfa1 <- fa(R,
            nfactors = 2,
            rotate = "none",
            fm = "ml" # only change
            )

mlfa1

## Factor Analysis using method =  ml
## Call: fa(r = R, nfactors = 2, rotate = "none", fm = "ml")
## Standardized loadings (pattern matrix) based upon correlation matrix
##            ML1   ML2   h2    u2 com
## ability   0.57  0.64 0.73 0.266 2.0
## parents   0.54  0.66 0.73 0.274 1.9
## teachers  0.49  0.64 0.66 0.344 1.9
## friends   0.42  0.56 0.49 0.509 1.9
## education 0.72  0.07 0.53 0.471 1.0
## college   1.00 -0.02 1.00 0.005 1.0
##
##                        ML1  ML2
## SS loadings           2.56 1.57
## Proportion Var        0.43 0.26
## Cumulative Var        0.43 0.69
## Proportion Explained  0.62 0.38
## Cumulative Proportion 0.62 1.00
##
## Mean item complexity =  1.6
## Test of the hypothesis that 2 factors are sufficient.
##
## df null model =  15  with the objective function =  3.3
## df of  the model are 4  and the objective function was  0.01
##
## The root mean square of the residuals (RMSR) is  0.01
## The df corrected root mean square of the residuals is  0.02
##
## Fit based upon off diagonal values = 1
## Measures of factor score adequacy
##                                                    ML1  ML2
## Correlation of (regression) scores with factors   1.00 0.91
## Multiple R square of scores with factors          1.00 0.83
## Minimum correlation of possible factor scores     0.99 0.67

## as before
mlfa1$loadings # without rotation

##
## Loadings:
##           ML1    ML2
## ability    0.575  0.635
## parents    0.535  0.663
## teachers   0.495  0.641
## friends    0.423  0.558
## education  0.723
## college    0.997
##
##                  ML1   ML2
## SS loadings    2.558 1.571
## Proportion Var 0.426 0.262
## Cumulative Var 0.426 0.688

mlfa1$communality

##   ability   parents  teachers   friends education   college
## 0.7336711 0.7255385 0.6555405 0.4907396 0.5286846 0.9950001

mlfa1$uniquenesses

##     ability     parents    teachers     friends   education     college
## 0.266328898 0.274461521 0.344459539 0.509260352 0.471315389 0.004999905

## finally factanal: ML estimation of factor analysis
## argument are a little different:
## x is a formula or a numeric matrix (not correlation matrix, it's the data!)
## facrors: n of factors wanted
## data: data for formula in x
## covmat = covariance matrix (if we dont' have the data)
## n.obs: is needed if we want scores or chisquare p-values
## rotation: default is "varimax", if we dont' want to rotate specichiamo "none"
## scores = non (se non vogliamo gli scores), regression(thomposon)or bartlet

mlfactanal1 <- factanal(factors = 2, covmat = R, n.obs = 556, rotation = 'none')
## again
mlfactanal1$loadings  # results exactly equal to fa with ml option

##
## Loadings:
##           Factor1 Factor2
## ability    0.575   0.635
## parents    0.535   0.663
## teachers   0.495   0.641
## friends    0.423   0.558
## education  0.723
## college    0.997
##
##                Factor1 Factor2
## SS loadings      2.558   1.571
## Proportion Var   0.426   0.262
## Cumulative Var   0.426   0.688
```

```
mlfactanal1$uniquenesses

##   ability   parents  teachers   friends education   college
## 0.2663356 0.2744603 0.3444599 0.5092609 0.4713136 0.0050000

1 - mlfactanal1$uniquenesses ## communality

##   ability   parents  teachers   friends education   college
## 0.7336644 0.7255397 0.6555401 0.4907391 0.5286864 0.9950000

## interesting: since we've specified n of obs we have chisq and p-value:
## p-value is larger than usual alpha: we can conclude that two factors are
## enough to explain correlation

rm(list = ls())
```

4.13 Example with dataset instead of correlation matrix

```
## spss
library(Hmisc)

##
## Caricamento pacchetto: 'Hmisc'
## Il seguente oggetto è mascherato da 'package:psych':
##
##     describe
## Il seguente oggetto è mascherato da 'package:lbmisc':
##
##     %nin%
## I seguenti oggetti sono mascherati da 'package:base':
##
##     format.pval, units

library(foreign)

Athletics <- spss.get("data/AthleticsData.sav")
dim(Athletics) # not a square matrix, not raw data

## [1] 1000    9

## measuring scores of 1000 individuals on some sports
corrplot(cor(Athletics))
```



```
## we see two blocks of variables with quite high correlation so it make sense
## to perform factor analysis
```

```
## we consider ML method
x = Athletics
(ml2 <- factanal(x, factors = 2, rotation = "none"))

##
## Call:
## factanal(x = x, factors = 2, rotation = "none")
##
## Uniquenesses:
##   PINBALL  BILLIARD      GOLF   X.1500M    X.2KROW X.12MINTR     BENCH      CURL
##     0.938     0.962     0.955     0.361     0.534     0.536     0.301     0.540
##  MAXPUSHU
##     0.560
##
## Loadings:
##           Factor1 Factor2
## PINBALL    0.249
## BILLIARD   0.192
## GOLF       0.206
## X.1500M            0.793
## X.2KROW    0.413   0.544
## X.12MINTR          0.681
## BENCH      0.813  -0.193
## CURL       0.673
## MAXPUSHU   0.545   0.379
##
##                Factor1 Factor2
## SS loadings      1.734   1.579
## Proportion Var   0.193   0.175
## Cumulative Var   0.193   0.368
##
## Test of the hypothesis that 2 factors are sufficient.
## The chi square statistic is 652.4 on 19 degrees of freedom.
## The p-value is 4.3e-126

## DIFFERENTLY here if we look at chisquare it's very high and p-value very
## low: we reject H_0 that our variables chan be explained by two factors.
## increase by 1 and see what happens
(ml3 <- factanal(x, factors = 3, rotation = "none"))

##
## Call:
## factanal(x = x, factors = 3, rotation = "none")
##
## Uniquenesses:
##   PINBALL  BILLIARD      GOLF   X.1500M   X.2KROW X.12MINTR     BENCH      CURL
##     0.635     0.414     0.455     0.361     0.520     0.538     0.302     0.536
##  MAXPUSHU
##     0.540
##
## Loadings:
##           Factor1 Factor2 Factor3
## PINBALL    0.425           0.429
## BILLIARD   0.443           0.624
## GOLF       0.447           0.585
## X.1500M            0.799
## X.2KROW    0.408   0.496  -0.260
## X.12MINTR          0.672
## BENCH      0.729  -0.280  -0.297
## CURL       0.605  -0.158  -0.270
## MAXPUSHU   0.512   0.317  -0.312
##
##                Factor1 Factor2 Factor3
## SS loadings      1.912   1.545   1.243
## Proportion Var   0.212   0.172   0.138
## Cumulative Var   0.212   0.384   0.522
##
## Test of the hypothesis that 3 factors are sufficient.
## The chi square statistic is 12.94 on 12 degrees of freedom.
## The p-value is 0.373

# ok p.value is larger and we don't reject h0 so 3 latent variables are ok for
# this dataset

## looking at the plot it seemed that two were enough, but loking better there
## are 3 blocks. corrplot is just a visual plot

ml3$uniquenesses
```

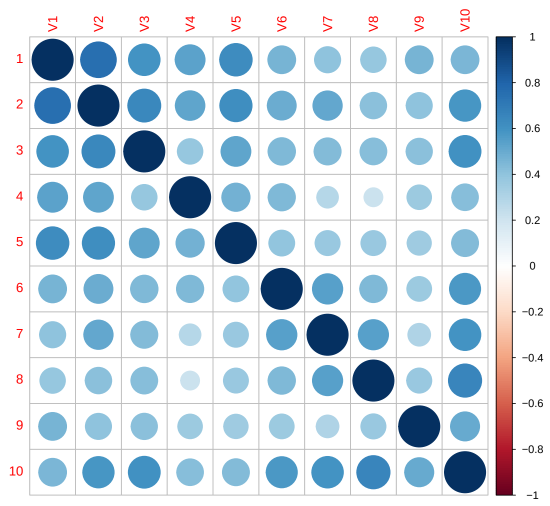
```
##   PINBALL  BILLIARD      GOLF   X.1500M   X.2KROW X.12MINTR     BENCH      CURL
## 0.6350926 0.4136649 0.4554615 0.3611385 0.5195340 0.5378475 0.3019471 0.5358651
##  MAXPUSHU
## 0.5396343

ml3$loadings

##
## Loadings:
##           Factor1 Factor2 Factor3
## PINBALL    0.425           0.429
## BILLIARD   0.443           0.624
## GOLF       0.447           0.585
## X.1500M            0.799
## X.2KROW    0.408   0.496  -0.260
## X.12MINTR          0.672
## BENCH      0.729  -0.280  -0.297
## CURL       0.605  -0.158  -0.270
## MAXPUSHU   0.512   0.317  -0.312
##
##                Factor1 Factor2 Factor3
## SS loadings      1.912   1.545   1.243
## Proportion Var   0.212   0.172   0.138
## Cumulative Var   0.212   0.384   0.522

##  if we don't consider any rotation is different to interpret the factors.try
##  to rotate
(ml4 <- factanal(x, factors = 3, rotation = "varimax"))

##
## Call:
## factanal(x = x, factors = 3, rotation = "varimax")
##
## Uniquenesses:
##   PINBALL  BILLIARD      GOLF   X.1500M   X.2KROW X.12MINTR     BENCH      CURL
##     0.635     0.414     0.455     0.361     0.520     0.538     0.302     0.536
##  MAXPUSHU
##     0.540
##
## Loadings:
##           Factor1 Factor2 Factor3
## PINBALL            0.131   0.590
## BILLIARD                   0.765
## GOLF                       0.735
## X.1500M    0.779  -0.179
## X.2KROW    0.585   0.372
## X.12MINTR  0.678
## BENCH     -0.119   0.816   0.137
## CURL               0.674
## MAXPUSHU   0.433   0.522
```

```
##
##                 Factor1 Factor2 Factor3
## SS loadings       1.613   1.584   1.502
## Proportion Var    0.179   0.176   0.167
## Cumulative Var    0.179   0.355   0.522
##
## Test of the hypothesis that 3 factors are sufficient.
## The chi square statistic is 12.94 on 12 degrees of freedom.
## The p-value is 0.373

ml4$loadings

##
## Loadings:
##           Factor1 Factor2 Factor3
## PINBALL            0.131   0.590
## BILLIARD                   0.765
## GOLF                       0.735
## X.1500M    0.779  -0.179
## X.2KROW    0.585   0.372
## X.12MINTR  0.678
## BENCH     -0.119   0.816   0.137
## CURL               0.674
## MAXPUSHU   0.433   0.522
##
##                 Factor1 Factor2 Factor3
## SS loadings       1.613   1.584   1.502
## Proportion Var    0.179   0.176   0.167
## Cumulative Var    0.179   0.355   0.522

## to ease interpretation is to use cutoff that are in absolute value larger
## than given

print(ml4$loadings, cutoff = 0.2, digits = 2)

##
## Loadings:
##           Factor1 Factor2 Factor3
## PINBALL                    0.59
## BILLIARD                   0.76
## GOLF                       0.73
## X.1500M    0.78
## X.2KROW    0.58    0.37
## X.12MINTR  0.68
## BENCH              0.82
## CURL               0.67
## MAXPUSHU   0.43    0.52
##
##                 Factor1 Factor2 Factor3
## SS loadings        1.61    1.58    1.50
## Proportion Var    0.18    0.18    0.17
## Cumulative Var    0.18    0.36    0.52

# interpretation becomes simpler: what we can see is tat the first factror has
# high entries for 1500M, 2KROW, 12mintr: it seems high ability in
# endurance/long sports variable

## second has high entries on the last three variables: it represent the
## strenght

## third factor has high entry on first three var: ability/coordination sport
## variable.

## this seems reasonable looking at correlation matrix
```

```
## ------------------------------------------
## --- now we extract the individual scores, having the dataset

## to do that
ml4reg <- factanal(x, factors = 3, rotation = "varimax", scores = "regression")
ml4bartlett <- factanal(x, factors = 3, rotation = "varimax", scores = "Bartlett")

## so now we have
head(ml4reg$scores)  # thompson: three factors

##          Factor1    Factor2     Factor3
## [1,] -0.64603135  0.5977488 -0.84052831
## [2,]  0.49077147 -0.1613787 -0.62070762
## [3,] -0.65306886 -0.9381925  0.73428198
## [4,] -0.02753535 -1.1794956  0.72268612
## [5,]  0.77251893  0.7731299  0.09496591
## [6,] -0.13301869 -0.0815587  0.38041785

head(ml4bartlett$scores) # bartlett: three factors

##          Factor1    Factor2     Factor3
## [1,] -0.81502246  0.8108272 -1.14286314
## [2,]  0.63677933 -0.1655105 -0.81955237
## [3,] -0.84449897 -1.2332453  1.04223966
## [4,] -0.04397501 -1.5387752  1.03323686
## [5,]  0.98562377  0.9715071  0.06573472
## [6,] -0.17546122 -0.1272071  0.51134257

## individuals which received highest score in sport about strenght

ord_scores <- order(ml4reg$scores[, 2], decreasing = TRUE) # second column is the str
head(ml4reg$scores[ord_scores, ])

##         Factor1  Factor2    Factor3
## [1,] -1.7772648 2.766277  1.1874676
## [2,]  0.4050792 2.603599  0.4903824
## [3,] -1.8201313 2.496708  0.6917442
## [4,]  1.4779385 2.339539 -0.5901477
## [5,]  1.0552924 2.279670  0.3561006
## [6,]  0.7579529 2.187697 -0.3514164
```

4.14 Last exercise

```
# intel dataset
x <- read.table("data/intel_test.txt") # it is a correlation matrix. do as the first
# exercise
R <- as.matrix(x)
R

##          V1    V2    V3    V4    V5    V6    V7    V8    V9   V10
##  [1,] 1.000 0.755 0.592 0.532 0.627 0.460 0.407 0.387 0.461 0.459
##  [2,] 0.755 1.000 0.644 0.520 0.617 0.497 0.511 0.417 0.406 0.583
##  [3,] 0.592 0.644 1.000 0.388 0.529 0.449 0.436 0.428 0.412 0.602
##  [4,] 0.532 0.528 0.388 1.000 0.475 0.442 0.280 0.214 0.361 0.424
##  [5,] 0.627 0.617 0.529 0.475 1.000 0.398 0.373 0.372 0.350 0.433
##  [6,] 0.460 0.497 0.449 0.442 0.398 1.000 0.545 0.446 0.366 0.575
##  [7,] 0.407 0.511 0.436 0.280 0.373 0.545 1.000 0.542 0.308 0.590
##  [8,] 0.387 0.417 0.428 0.214 0.372 0.446 0.542 1.000 0.375 0.654
##  [9,] 0.461 0.406 0.412 0.361 0.355 0.366 0.308 0.375 1.000 0.502
## [10,] 0.459 0.583 0.602 0.424 0.433 0.575 0.590 0.654 0.502 1.000

## 75 children in intelligence test
## V1: information
## V2 vocabulary
## ....

## è lo stesso dataset visto con la prof?

corrplot(R)
```



```
(ml1 <- factanal(covmat = R, factors = 2, n.obs = 75, rotation = 'none'))

##
## Call:
## factanal(factors = 2, covmat = R, n.obs = 75, rotation = "none")
##
## Uniquenesses:
##    V1    V2    V3    V4    V5    V6    V7    V8    V9   V10
## 0.215 0.249 0.452 0.622 0.482 0.553 0.534 0.481 0.679 0.177
##
## Loadings:
##       Factor1 Factor2
##  [1,]  0.789  -0.403
##  [2,]  0.834  -0.234
##  [3,]  0.740
##  [4,]  0.587  -0.185
##  [5,]  0.676  -0.247
##  [6,]  0.654   0.140
##  [7,]  0.641   0.235
##  [8,]  0.630   0.351
##  [9,]  0.564
## [10,]  0.807   0.414
##
##                Factor1 Factor2
## SS loadings      4.872   0.685
## Proportion Var   0.487   0.069
## Cumulative Var   0.487   0.556
##
## Test of the hypothesis that 2 factors are sufficient.
## The chi square statistic is 16.51 on 26 degrees of freedom.
## The p-value is 0.923
```

```
# p.val very high: two factors are enough to explain correlation of this data

ml1$uniquenesses

##        V1        V2        V3        V4        V5        V6        V7        V8
## 0.2147868 0.2492602 0.4516243 0.6217758 0.4822172 0.5528077 0.5338987 0.4806868
##        V9       V10
## 0.6790751 0.1769536

ml1$loadings

##
## Loadings:
##       Factor1 Factor2
##  [1,]  0.789  -0.403
##  [2,]  0.834  -0.234
##  [3,]  0.740
##  [4,]  0.587  -0.185
##  [5,]  0.676  -0.247
##  [6,]  0.654   0.140
##  [7,]  0.641   0.235
##  [8,]  0.630   0.351
##  [9,]  0.564
## [10,]  0.807   0.414
##
##                Factor1 Factor2
## SS loadings      4.872   0.685
## Proportion Var   0.487   0.069
## Cumulative Var   0.487   0.556
```

when we dont' perform rotation: the first factor has high entries for all the variables (its an average measurmente) while the other are contrasts.

```
(ml2 <- factanal(covmat = R, factors = 2, n.obs = 75, rotation = 'varimax'))

##
## Call:
## factanal(factors = 2, covmat = R, n.obs = 75, rotation = "varimax")
##
## Uniquenesses:
##    V1    V2    V3    V4    V5    V6    V7    V8    V9   V10
## 0.215 0.249 0.452 0.622 0.482 0.553 0.534 0.481 0.679 0.177
##
## Loadings:
##       Factor1 Factor2
##  [1,] 0.852   0.245
##  [2,] 0.769   0.399
##  [3,] 0.563   0.481
##  [4,] 0.555   0.266
##  [5,] 0.662   0.281
##  [6,] 0.382   0.549
##  [7,] 0.308   0.609
##  [8,] 0.220   0.686
##  [9,] 0.375   0.424
## [10,] 0.307   0.854
##
##                Factor1 Factor2
## SS loadings      2.904   2.653
## Proportion Var   0.290   0.265
## Cumulative Var   0.290   0.556
##
## Test of the hypothesis that 2 factors are sufficient.
## The chi square statistic is 16.51 on 26 degrees of freedom.
## The p-value is 0.923
```

```
print(ml2$loadings) # first factor high on first five variables, second on last
```

```
##
## Loadings:
##       Factor1 Factor2
##  [1,] 0.852   0.245
##  [2,] 0.769   0.399
##  [3,] 0.563   0.481
##  [4,] 0.555   0.266
##  [5,] 0.662   0.281
##  [6,] 0.382   0.549
##  [7,] 0.308   0.609
##  [8,] 0.220   0.686
##  [9,] 0.375   0.424
## [10,] 0.307   0.854
##
##                Factor1 Factor2
## SS loadings      2.904   2.653
## Proportion Var   0.290   0.265
## Cumulative Var   0.290   0.556
```

```
# five variables
# vedi interpretazione poi negli appunti
```

Capitolo 5

Discriminant analysis and supervised classification

Goals are different, but methods developed for discriminat anaysis can be used for supervised classification.
Final goal of supervised classification is to predict a group on data for which the group is unknown. Differently from the unsupervised case, we know which are the elemnts from one groups or the other.
Our plan is see:

1. how discriminant analysis works
2. supervised classification

5.1 Discriminant analysis

It's an old method invented by Fisher which still now works very well. Advantage wrt to other machine learning is that it's interpretable.
Methods is based on *linear combination*: find a linear combination that allows us to discriminate between groups. If we use a neural network to do the same it all become very less clear.

Suppose we have two population, Π_1 and Π_2. I observe a variable X $(p \times 1)$ in both which will have

- mean μ_1, μ_2 in the two population
- variance/covariance $\Sigma_1 = \Sigma_2 = \Sigma$: only assumptions Fisher made was that the two population have the same variance, that are homoscedastic.

From the two population we obtain a samples C_1 and C_2, composed by n_1 units and n_2 units respectively. the observed variable X has:

- mean $\overline{x}_1$ and $\overline{x}_2$ in the two groups
- covariance S_1 and S_2 in the second sample (we assume that the two estimates of varcov are the unbiased one but it is not necessary)

Example 5.1.1 (swiss banknotes)**.** classification of true and false banconotes, basata su margin top e margin bottom (dimensions), in the image red points are the true banknotes, blue are fake.
groups are spearated but if we project points on the dimension X the two groups are overapping, while on the Y is slighltly better.
Can I find a linear combination of two variables which allows me to separate better the two groups?

TODO: grafico?

Something that does better than original variables is the linear combination where the groups are best separated and variances homogeneous.
Fisher proposed to look for the linear combination of the observed variables such that, when projected along it, the groups are maximally separated (distant means) and at the same time maximally homogeneous (variance within group is small).
He wanted to find a such as

$$\mathbf{Y} = \mathbf{a}^T\mathbf{X}$$

the groups in y are maximally separated and homogeneous.
Fisher developed Anova which is based on the ratio of the between group sum of square and within group sum of square. The idea of LDA is to find **a** such that when projected, variance of Y between is maximum and variance Y within is minimum. This idea/method was consistent with the idea of Anova.

TODO: CHECK

First we need to define the means:

- the mean of y in group 1 is a linear combination of mean of original data

$$\overline{y}_1 = a^T\overline{X}_1$$
$$\overline{y}_2 = a^T\overline{X}_2$$

- the overall mean is

$$\overline{y} = \frac{n_1\overline{y}_1 + n_2\overline{y}_2}{n_1+n_2} = \frac{n_1a^t\overline{x}_1 + n_2a^T\overline{x}_2}{n}$$

- the between variance of y is the between group sum of square divided by the degrees of freedom (here G is the general number of groups, here we have two so $G = 2$)

$$V(y) = \frac{BSS}{df} = \frac{\sum_{g=1}^{G}(\overline{y}_g - \overline{y})^2 \cdot n_g}{G-1}$$

the degrees of freedom are due to G means and 1 constraint (1 grand mean): in the two group cases, $df = 1$. So going on with two groups it simplify to

$$\begin{aligned} V(y) &= \sum_{g=1}^{2}(\overline{y}_g - \overline{y})^2 \cdot n_g = \left(a^T\overline{x}_1 - a^T\overline{x}\right)^2 \cdot n_1 + \left(a^T\overline{x}_2 - a^T\overline{x}\right)^2 \cdot n_2 \\ &= \left[a^T(\overline{x}_1 - \overline{x})\right]^2 n_1 + \left[a^T(\overline{x}_2 - \overline{x})\right]^2 n_2 \\ &= a^T(\overline{x}_1 - \overline{x})(\overline{x}_1 - \overline{x})^T a n_1 a^T(\overline{x}_2 - \overline{x})(\overline{x}_2 - \overline{x})^T a n_2 \\ &\overset{(=)}{1}\, a^T\left[(\overline{x}_1 - \overline{x})(\overline{x}_1 - \overline{x})^T n_1 + (\overline{x}_2 - \overline{x})(\overline{x}_2 - \overline{x})^T n_2\right] a \end{aligned}$$

where in (1) we collected a^T and a at the end. In general terms (for G unspecified) the above becomes

$$a^T\left[\sum_{g=1}^{G}(\overline{x}_g - \overline{x})(\overline{x}_g - \overline{x})^T \cdot n_g\right] a$$

and in general with G groups we have that the between variance of the linear combination is

$$V(y) = a^T \underbrace{\frac{\left[\sum_{g=1}^{G}(\overline{x}_g - \overline{x})(\overline{x}_g - \overline{x})^T \cdot n_g\right]}{G-1}}_{\mathbf{B}} a_T \tag{5.1}$$

where $\mathbf{B}$ is the between group variance in the original X space. I can write the variance of Y between as

$$V(y) = \mathbf{a}^T\mathbf{B}\mathbf{a}$$

We can find an alternative way to write $\mathbf{B}$, in case we have only two groups[1]. Starting back to $V(Y)$

$$\begin{aligned} V(y) &= \sum_{g=1}^{2}(\overline{y}_g - \overline{y})^2 \cdot n_g = (\overline{y}_1 - \overline{y})^2 \cdot n_1 + (\overline{y}_2 - \overline{y})^2 \cdot n_2 \\ &= n_1\left[a^T\overline{x}_1 - \frac{1}{n}\left(a^T\overline{x}_1 n_1 + a^T\overline{x}_2 n_2\right)\right]^2 + n_2\left[a^T\overline{x}_2 - \frac{1}{n}\left(a^T\overline{x}_1 n_1 + a^T\overline{x}_2 n_2\right)\right]^2 \\ &= \frac{n_1\left[na^T\overline{x}_1 - a^T\overline{x}_1 n_1 - a^T\overline{x}_2 n_2\right]^2}{n^2} + n_2\frac{\left[na^T\overline{x}_2 - a^T\overline{x}_1 n_1 - a^T\overline{x}_2 n_2\right]^2}{n^2} \\ &= \frac{n_1}{n^2}\left(a^T\overline{x}_1 n_2 - a^T\overline{x}_2 n_2\right)^2 + \frac{n_2}{n^2}\left(a^T\overline{x}_2 n_1 - a^T\overline{x}_1 n_1\right)^2 \\ &= \frac{n_1 n_2^2}{n}\left[a^T(\overline{x}_1 - \overline{x}_2)\right]^2 + \frac{n_2 n_1^2}{n^2}\left[a^T(\overline{x_2} - \overline{x}_1)\right]^2 \\ &= \frac{n_1 n_2^2}{n^2} a^T(\overline{x}_1 - \overline{x}_2)(\overline{x}_1 - \overline{x}_2)^T a + \frac{n_1^2 n_2}{n^2} a^T(\overline{x}_1 - \overline{x}_2)(\overline{x}_1 - \overline{x}_2)^T a \\ &= \frac{n_1 n_2^2 + n_1^2 n_2}{n^2} a^T(\overline{x}_1 - \overline{x}_2)(\overline{x}_1 - \overline{x}_2)^T a \\ &\overset{(1)}{=} \frac{n_1 \cdot n_2}{n_1 + n_2} a^T(\overline{x}_1 - \overline{x}_2)(\overline{x}_1 - \overline{x}_2)^T a \end{aligned}$$

[1]Fisher based it on IRIS flower dataset (three species, he wanted to find which caracteristics of the iris was able to sperate in the three groups of iris).

where in (1) we simplified using $n^2 = (n_1 + n_2)^2$.
So in the two groups cases here we have that

$$\mathbf{B} = \frac{n_1 \cdot n_2}{n_1 + n_2}(\overline{x}_1 - \overline{x}_2)(\overline{x}_1 - \overline{x}_2)^T$$

this is a way computing **B** for just two groups.
Now regarding B:

- Dimension of matrix **B**: with p variables B is a $p \times p$ matrix
- what's its rank? **B** is the product of the col vector times row vector: the product is rank 1 (since both row and col vctor has rank 1). So in case of in the case of two group **B** has rank 1.
  In the general case has at most rank G - 1

TODO: CHECK

Thus the matrix B **is not full rank**.

Now let's check Variance within, that is the sum of variances withing group. In the two group cases, using the unbiased estimator we have the weighted mean of variances that is

$$V(y) = \frac{a^T S_1 a \cdot (n_1 - 1) + a^T S_2 a \cdot (n_2 - 1)}{n_1 + n_2 - 2}$$

using unbiased estimator we have that $a^T S_1 a$ is the variance of y in the group 1 which has to be weighted by the degrees of freedom (take $n_1 - 1$ for teh unbiased version).
We can rewrite as

$$V(y) = a^T \underbrace{\left[\frac{S_1(n_1 - 1) + S_2(n_2 - 1)}{n - 2}\right] a}_{\mathbf{W}}$$

where **W** is the within group variance in the X space so

$$V(y) = a^T W a$$

In the general G group case the within group sum of square in the X spaceis

$$\mathbf{W} = \frac{\sum_{g=1}^{G}(n_g - 1)S_g}{n - G}$$

TODO: CHECK

we have done something like this when performing Student's t-test. Now we rephrased within and between variance in terms of a so we can choose a optimizing.

Important remark 16 (recappino). only assumption so far is that the two population shares covariance matrix (not any hypothesis on shape of distribution). find a such that when projected the two groups are as much separated finding the direction amounts to find a linear cut to the groups.
we have what needed to derive fisher criterion:

$$V(Y)_b = a^T \mathbf{B} a$$
$$V(Y)_w = a^T \mathbf{W} a$$

with **B** and **W** the between and within group variance in the x space.

Fisher wanted to find a linear combination $y = a^T X$ such that the ratio

$$\varphi = \frac{V(y)_b}{V(y)_w}$$

is maximum with respect to a. He wanted mostly separated (max Variance at the numerator num) and homogeneous (min Variance denominator). Let's write ϕ as

$$\varphi = \frac{a^T B a}{a^T W a}$$

our problem is

$$\max_a \varphi$$

is a standard maximum problem so we derive wrt to a and equate to 0

$$\begin{aligned}
\frac{\partial \varphi}{\partial \mathbf{a}} &= \frac{2Ba(a^T W a) - 2Wa(a^T B a)}{(a^T W a)^2} \\
&= 2\left[\frac{Ba(a^T W a)}{(a^T W a)^2} - \frac{Wa(a^T B a)}{(a^T W a)^2}\right] \\
&\overset{(=)}{1} 2\left[\frac{Ba}{a^T W a} - \frac{Wa\phi}{a^T W a}\right]
\end{aligned}$$

where in (1) we just renamed the constant $\frac{a^T B a}{a^T W a} = \phi$. Furthermore $a^T W a$ is a constant $\neq 0$; by equating to 0 for maximization we can simplify to

$$\begin{aligned}
Ba - \phi W a &= 0 \\
(B - \phi W)a &= 0
\end{aligned}$$

Now we premultiply both sides by W^{-1}

$$\big(\mathbf{W}^{-1}\mathbf{B} - \phi \mathbf{I}\big)\mathbf{a} = \mathbf{0}$$

This last is a linear equation system which admits a nontrivial solution iff the determinant

$$\det \mathbf{W}^{-1}\mathbf{B} - \phi \mathbf{I} = 0$$

that is if ϕ is a root of the characteristics polynomial of $\mathbf{W}^{-1}\mathbf{B}$. This means that ϕ is an eigenvalue of $W^{-1}B$ and $\mathbf{a}$ is the corresponding eigenvector. We're again trying to solve a linear equationss system as done for principal components.
If we prefer we can rewrite as the eigenvalue eigenvector relationship

$$(W^{-1}B - \phi \mathbf{I})a = 0 \implies W^{-1}Ba = \phi a$$

Some differences with Principal components:

1. here we don t need any constraint (there we needed to constraint a to have norm 1): here we dont need constraint because it appears both at numerator and denominator

2. while the covariance matrix S, Σ is symmetric, this matrix $\mathbf{W}^{-1}\mathbf{B}$ is non-symmetric (it's the produt of two symmetric matrices but it is not). Spectral decomposition exists but there can be immaginary eigenvalue/vectors

Important remark 17. How many non zero eigenvalue do we have?

- the rank of $\mathbf{W}^{-1}\mathbf{B}$: the rank of W is p in two group cases, also B is $p \times p$ matrix but in the two group cases its rank is 1. So the product $\mathbf{W}^{-1}\mathbf{B}$ will have rank 1. Therefore it will have just 1 non zero eigenvalue.
  In the 2 group cases the **best linear discriminant direction** will be the *eigenvector* of $W^{-1}B$ corresponding to the non zero eigenvalue.

- in the general G group case the rank $\mathbf{W}^{-1}\mathbf{B}$ will be at most of rank $G-1$: this implies that it will have at most $G-1$ non null eigenvalue and at most $G-1$ discriminant directions
  The optimal linear discriminant direction will be defined by the *eigenvector* of $W^{-1}B$ corresponding to the *largest eigenvalue* as ϕ.
  However in the general $G > 2$ case we can consider more than 1 discriminant directions that corresponds to different eigenvalues in decreasing order. This means that for instance three groups can be optimally separated by just two discriminant directions, independently from how large p is (eg I can have 100 variables). eg if we have three groups we can go on and find the second discriminant direction corresponding to the second highest larger eigenvalue.
  assuming we have three groups like fig **??** maybe a single direction is enough, but if the situa is the one depicted in **??** two directions are needed

TODO: make figures

Example 5.1.2 (Fisher Iris Data)**.** we have data on 150 iris flowers belonging to three species (setosa versicolor and verginica). for each flower the legth and width of the flower and sepal length and width ($x_1^T, \dots x_3^T$ are the three groups means)
we have three groups, we expect to find two discriminant directions:

- looking at the variables: the first direction separates flower with big sepal and low petal and viceversa

- the second direction a_2^T is focused on sepal only

the projection on LD1 and LD2: setosa è + per i cazzi suoi sulla prima componente, sulla seconda non vi è troppa separazione. La separazione su LD1 è buona.

Linear discriminant functions The linear combinations:

$$\begin{aligned} y_1 &= \mathbf{a}_1^T X \\ y_2 &= \mathbf{a}_2^T X \\ &\dots \\ y_{G-1} &= a_{G-1}^T X \end{aligned}$$

with $\mathbf{a}_1, ldots, \mathbf{a}_{G-1}$ are the eigenvectors corresponding to the non zero eigenvalues in decreasing order, are called linear discriminant functions (or also canonical variates).

Lets' consider two different discriminant directions $\mathbf{a}_i, \mathbf{a}_j$. We have that from the equation it must hold that:

$$B\mathbf{a}_i = \phi_i \mathbf{W}\mathbf{a}_i$$
$$B\mathbf{a}_j = \phi_j \mathbf{W}\mathbf{a}_j$$

Let's pre multiply the first by $\mathbf{a}_j^T$ and the second by $\mathbf{a}_i^T$

$$\mathbf{a}_j^T \mathbf{B}\mathbf{a}_i = \phi_i \mathbf{a}_j^T \mathbf{W}\mathbf{a}_i$$
$$\mathbf{a}_i^T \mathbf{B}\mathbf{a}_j = \phi_j \mathbf{a}_i^T \mathbf{W}\mathbf{a}_j$$

In the left hand side of the two equations we have two quantities that are the same: two scalar which is one the transpose of the other (transpose of scalar is the scalar itself), that is

$$\mathbf{a}_j^T \mathbf{B}\mathbf{a}_i = \mathbf{a}_i^T \mathbf{B}\mathbf{a}_j$$

thus event the right hand side has to be the same

$$\phi_i \mathbf{a}_j^T \mathbf{W}\mathbf{a}_i = \phi_j \mathbf{a}_i^T \mathbf{W}\mathbf{a}_j$$

and even $\mathbf{a}_j^T \mathbf{W}\mathbf{a}_i = \mathbf{a}_i^T \mathbf{W}\mathbf{a}_j$ for the same reason before.
If ϕ_i and ϕ_j are different from zero, and different each other, in order for the equality to hold it must be that

$$\mathbf{a}_j^T \mathbf{W}\mathbf{a}_i = \mathbf{a}_i^T \mathbf{W}\mathbf{a}_j = 0$$

Now if we put all the discriminant direction together in the matrix $\mathbf{A}$ we have that

$$\mathbf{A}^T \mathbf{W}\mathbf{A} \quad \text{is diagonal}$$

because crossproducts are all equal to 0.
This means that **discrimnant directions are uncorrelated within group**: as $\mathbf{a}_i^T \mathbf{W}\mathbf{a}_j$ is the covariance between y_i and y_j within group (there's $\mathbf{W}$), then we conclude that the linear discriminant functions are uncorrelated within groups.

TODO: CHECK

before we said that

$$a_i^T W a_j = 0 \implies a_i^T B a_j = 0$$

so it means that the discriminant function are **uncorrelated also between groups**.
If they are uncorrelated both within and between, they are overall ncorrelated so this means that $\mathbf{A}^T \mathbf{S}\mathbf{A}$ is diagonal.
This is one important difference we'have not asked for: the orthogonality.

5.2 Doing classification

How can we use the results obtained so far in order to perform classification? Classification is the process of assigning a unit whose group membership is unknown to one of set of G groups.
How to use the linear discriminant directions? Let's assume we're in the two groups cases: i have data x_0 on a new unit whose membership is unknown (a $p \times 1$ vector) with the same variables i have used to obtain the discriminant function. I can calculate

$$y_0 = \mathbf{a}^T x_0$$

where **a** is the discriminant direction. In the two group case i also compute the projection of the mean of groups along **a**

$$\overline{y}_1 = \mathbf{a}^T \overline{x}_1$$
$$\overline{y}_2 = \mathbf{a}^T \overline{x}_2$$

Let's assume for simplicity that $\overline{y}_1 > \overline{y}_2$

To classify my unit i project it on the discriminant direction (calculating y_0) and if

$$|y_0 - \overline{y}_2| < |y_0 - \overline{y}_1| \implies x_0 \in \Pi_2$$

i assign to the second population. Otherwise if

$$|y_0 - \overline{y}_1| < |y_0 - \overline{y}_2| \implies x_0 \in \Pi_1$$

i assign to the first one. In general

$$\begin{aligned} y_0 &> \frac{\overline{y}_1 + \overline{y}_2}{2} \implies x_0 \in \Pi_1 \\ y_0 &< \frac{\overline{y}_1 + \overline{y}_2}{2} \implies x_0 \in \Pi_2 \\ y_0 &= \frac{\overline{y}_1 + \overline{y}_2}{2} \implies \text{toss a coin to decide which group} \end{aligned}$$

Fisher proved that in the two group case, the eigenvector of $\mathbf{W}^{-1}\mathbf{B}$ corresponding to the only non zero eigenvalues can be obtained in closed form as

$$a = \mathbf{W}^{-1}(\overline{x}_1 - \overline{x}_2)$$

TODO: CHECK

By exploiting this we can say

$$y = a^T X = (\overline{x}_1 - \overline{x}_2)^T W^{-1} X$$

where being **W** symmetric, it's transpose coincides with the inverse. The group transformed means become

$$\overline{y}_1 = a^T \overline{x}_1 = (\overline{x}_1 - \overline{x}_2)^T W^{-1} \overline{x}_1$$
$$\overline{y}_2 = a^T \overline{x}_2 = (\overline{x}_1 - \overline{x}_2)^T W^{-1} \overline{x}_2$$

So the rule to assign x_0 in Π_1, becomes

$$y_0 > \frac{\overline{y}_1 + \overline{y}_2}{2} \tag{5.2}$$

$$(\overline{x}_1 - \overline{x}_2)^T W^{-1} X_0 > \frac{1}{2}(\overline{x}_1 - \overline{x}_2)^T W^{-1}(\overline{x}_1 + \overline{x}_2) \tag{5.3}$$

This last is called **Fisher 's linear discriminant** classification rule.

Remark 11. We'll see that the classification problem can be addressed directly in the original p-dimensional space by assigning a unit whose group membership is unknown to the population from which it has the smallest Mahalanobis distance

MALAHNOBIS: special kind, actually weighted, euclidean distance mahalanomis distance is just a weighted euclidean distance (which consider the covariance/correlation of two variables). **TODO**: inserire figura

The distance of a unit with the mean of the first group is

$$(x_0 - \overline{x_1})^T(x_0 - \overline{x_1}) = x_0 - \overline{x_1})^T \mathbf{I}(x_0 - \overline{x_1}) \quad \text{(squared) eulidean d}$$

$$x_0 - \overline{x_1})^T \mathbf{W}^{-1}(x_0 - \overline{x_1}) \quad \text{Mahalanobis distance}$$

The only difference above is the $\mathbf{I}$ vs $\mathbf{W}^{-1}$: actually Mahalanobis is a weighted euclidean distance, has an inverse covariance matrix in between.
Considering the distance between the unit and the second group mean this is

$$x_0 - \overline{x_2})^T \mathbf{W}^{-1}(x_0 - \overline{x_2})$$

So the rule to assign to Π_1 becomes

$$x_0 - \overline{x_1})^T \mathbf{W}^{-1}(x_0 - \overline{x_1}) < x_0 - \overline{x_2})^T \mathbf{W}^{-1}(x_0 - \overline{x_2})$$
$$x_0^T W^{-1} x_0 - \overline{x}_1^T W^{-1} x_0 - x_0^T W^{-1} \overline{x}_1 + \overline{x}_1^T W^{-1} \overline{x}_1 < x_0^T W^{-1} x_0 - x_2^T W^{-1} x_0 - x_0^T W^{-1} \overline{x}_2 + \overline{x}_2^T W^{-1} \overline{x}_2$$

now we simplify the first two terms in both terms of inequality and note that $\overline{x_1}^T W^{-1} x_0$ is the transpose of $x_0 W^{-1} \overline{x}_1$ so we put together, as well as $\overline{x}_2^T W^{-1} x_0$ and $x_0^T W^{-1} \overline{x}_2$ are:

$$-2\overline{x}_1 W^{-1} x_0 + \overline{x}_1^T W^{-1} \overline{x}_1 < -2\overline{x}_2 W^{-1} x_0 + \overline{x}_2^T W^{-1} \overline{x}_2$$
$$2\overline{x}_1^T W^{-1} x_0 - 2\overline{x}_2 W^{-1} x_0 > \overline{x}_1^T W^{-1} \overline{x}_1 - x_2 W^{-1} \overline{x}_2$$

at the second member we have a difference of two squares. We end with the same rule as before

$$(\overline{x}_1 - \overline{x}_2)^T W^{-1} x_0 > \frac{1}{2}(\overline{x}_1 - \overline{x}_2)^T W^{-1}(\overline{x}_1 - \overline{x}_2)$$

Important remark 18. So assigning a unit to the population it is closest to in the linear discriminant space is the same as assigning the unit to the populationo which it has the smallest mahalanobis distance in the original p-dimensional space.

Remark 12. one final comment: in the linear discriminant space measure distances using the euclidean distance because the linear discriminant function are uncorrelated (so we don't need a weight that accounts for correlation)

using mahalanobis is original space is same as using euclidean in the transformed space. **TODO**: CHECK

5.3 Classification based on probability models

We can address classification directly: it can be done in a more precise way knowing something more from the population.
We know the density of the p-dimensional vector of covariate X and assuming we have x_0 an unit whose group membership is unknown and Π_1, Π_2 populations. As said, we assume to know the density of the unit if it comes from either one of the two populations

$$f(x|x \in \Pi_1) = f_1(x)$$
$$f(x|x \in \Pi_2) = f_2(x)$$

We call R the sample space: set of all the possible values that X can take. The goal is to split R into two regions (mutually exclusive and exaustive), R_1 and R_2, in such that if x_0 falls in R_1 it is assigned to population Π_1, and if x_0 falls in R_2 it is assigned to Π_2 and the probability of a wrong assignment is minimum

TODO: CHECK

I want to find the split in R_1 and R_2 that when applying it to classify. We're measuring probabilities now, we want to minimize probability of wrong classification.
To understand, assume we've two population with density: population are not perfectly separated, tails overlap, the two population shares a part of the domain.

TODO: fig

we want to split the domain R:

- in case of the x_0 on the figure it is more likely that the unit comes from Π_1 than Π_2. so I assign it to Π_1
- in case of x_2 it is more liklely that the unit comes from Π_2 than from Π_1, so I assign it to Π_2
- the cutoff between the two population will be at the cross of the two curves in yellow. the split is with R_1 to the left and R_2 on the right

What done so far can be translated in a very simple calssfiication rule

$$\frac{f_1(x_0)}{f_2(x_0)} > 1 \implies x_0 \in \Pi_1$$

because it's more likely that x_0 comes from population 1 than if it comes from population 2. So R_1 is defined as the set of x values sucht that

$$R_1 = \left\{ x : \frac{f_1(x_0)}{f_2(x_0)} > 1 \right\}$$
$$R_2 = \left\{ x : \frac{f_1(x_0)}{f_2(x_0)} < 1 \right\}$$
$$\text{toss a coin if } \left\{ x : \frac{f_1(x_0)}{f_2(x_0)} = 1 \right\}$$

Assuming i know the distribution, i compute the likelihood of the data, i compute the ratio of likelihood and then choose.

This rule is **likelihood ratio rule** (f_1 and f_2 plays the role of likelihood)
However things are not always so easy: not all population are equally likely (consider the prior): not all the population have the same a priori probability (eg rare disease common disease we need to take in account). Now hypothesize:

- π_1 to be the prior probability that a unit comes from population 1
- π_2 to be the prior probability that a unit comes from population 2
- $\pi_1 + \pi_2 = 1$

THe total probability of a wrong classification p is

$$p = p(1|2) + p(2|1)$$

that is the sum of probability that I assign a unit to population 1 when it comes from population 2 and the viceversa. this two kind of mistake could have different probs. How to define this probability?

$$\begin{aligned} p(2|1) &= \pi_1 \int_{R_2} f_1(x)\,\mathrm{d}x \\ p(1|2) &= \pi_2 \int_{R_1} f_2(x)\,\mathrm{d}x \end{aligned}$$

so the total probability of a wrong classification becomes

$$p = \pi_1 \int_{R_2} f_1(x)\,\mathrm{d}x + \pi_2 \int_{R_1} f_2(x)\,\mathrm{d}x$$

We want to find R_1 and R_2 so that this quantity is minimum. To find the solution we need to remember that

$$\int_R f_1(x)\,\mathrm{d}x = 1, \quad \int_R f_2(x)\,\mathrm{d}x = 1,$$
$$R_1 \cup R_2 = R, R_1 \cap R_2 = \emptyset$$

Thus considering the first density can be split in the two areas

$$\int_R f_1(x)\,\mathrm{d}x = \int_{R_1} f_1(x)\,\mathrm{d}x + \int_{R_2} f_1(x)\,\mathrm{d}x = 1$$

So the quantity i'm interested in is

$$\int_{R_2} f_1(x)\,\mathrm{d}x = 1 - \int_{R_1} f_1(x)\,\mathrm{d}x$$

we pick this above and go back to the total probability of wrong classification which becomes

$$\begin{aligned} p &= \pi_1 \int_{R_2} f_1(x)\,\mathrm{d}x + \pi_2 \int_{R_1} f_2(x)\,\mathrm{d}x \\ &= \pi_1 \left[1 - \int_{R_1} f_1(x)\,\mathrm{d}x\right] + \pi_2 \int_{R_1} f_2(x)\,\mathrm{d}x \\ &= \pi_1 - \pi_1 \int_{R_1} f_1(x)\,\mathrm{d}x + \pi_2 \int_{R_1} f_2(x)\,\mathrm{d}x \\ &= \pi_1 - \int_{R_1} \pi_2 f_2(x) - \pi_1 f_1(x)\,\mathrm{d}x \end{aligned}$$

we minimize this: the integral is minimum for the values of x for which the integrand is negative!

$$\begin{aligned} R_1 &= \{x : \pi_2 f_2(x) - \pi_1 f_1(x) < 0\} \\ &= \{x : \pi_1 f_1(x) > \pi_2 f_2(x)\} \\ &= \left\{x : \frac{f_1(x)}{f_2(x)} > \frac{\pi_2}{\pi_1}\right\} \end{aligned}$$

In the last one the first inequality member is a likelihood ratio as seen before and the second the switched ratio on priors probability.
This is the rule minimizing the total probability of a wrong classification: likelihood ratio are compared with inverse priors ratio. If $\pi_1 = \pi_2 = 1/2$, then the rule minimizing the total probability of a wrong classification coincides with the likelihood ratio rule seen before, otherwise results are different and take into account the fact theat different population have different prior probabilities.

The same result can be obtained using Bayes theorem. For the following:

- $\mathbb{P}(X \in \Pi_1 | X = x_0)$ is the posterior probability for a observed unit to belong to population Π_1
- $\mathbb{P}(X = x_0 | X \in \Pi_1) = f(X = x_0 | X \in \Pi_1)$ is the likelihood of a single observation

we have that the posterior probability for population 1 and 2 can be rewritten according to Bayes rule as

$$\mathbb{P}(X \in \Pi_1 | X = x_0) = \frac{\mathbb{P}(X \in \Pi_1) \cdot \mathbb{P}(X = x_0 | X \in \Pi_1)}{\mathbb{P}(X \in \Pi_1) \cdot \mathbb{P}(X = x_0 | X \in \Pi_1) + \mathbb{P}(X \in \Pi_2) \cdot \mathbb{P}(X = x_0 | X \in \Pi_2)}$$

$$\mathbb{P}(X \in \Pi_2 | X = x_0) = \frac{\mathbb{P}(X \in \Pi_2) \cdot \mathbb{P}(X = x_0 | X \in \Pi_2)}{\mathbb{P}(X \in \Pi_1) \cdot \mathbb{P}(X = x_0 | X \in \Pi_1) + \mathbb{P}(X \in \Pi_2) \cdot \mathbb{P}(X = x_0 | X \in \Pi_2)} = 1 - \mathbb{P}(X$$

I can assign a unint to a population which have the largest posterior probability of coming from. the ensuing rule is called the optimal bayes's rule.

$$\frac{\mathbb{P}(X \in \Pi_1) \cdot \mathbb{P}(X = x_0 | X \in \Pi_1)}{\mathbb{P}(X \in \Pi_1) \cdot \mathbb{P}(X = x_0 | X \in \Pi_1) + \mathbb{P}(X \in \Pi_2) \cdot \mathbb{P}(X = x_0 | X \in \Pi_2)} > \frac{\mathbb{P}(X \in \Pi_2) \cdot \mathbb{P}}{\mathbb{P}(X \in \Pi_1) \cdot \mathbb{P}(X = x_0 | X \in \Pi_1}$$

To simplify notation substitute the priors, eg $\mathbb{P}(X \in \Pi_1)$ with π_1 and likelihoods, eg $\mathbb{P}(X = x_0 | X \in \Pi_2)$ with $f_2(x_0)$. Thus the rule becomes

$$\frac{\pi_1 f_1(x_0)}{\pi_1 f_1(x_0) + \pi_2 f_2(x_0)} > \frac{\pi_2 f_2(x_0)}{\pi_1 f_1(x_0) + \pi_2 f_2(x_0)} \qquad \pi_1 f_1(x_0) > \pi_2 f_2(x_0)$$

$$\frac{f_1(x_0)}{f_2(x_0)} > \frac{\pi_2}{\pi_1}$$

which ends to be the same rule found before: if above holds we assign to Π_1 otherwise to ¶$_2$. The classification rule minimizing the total probability of a wrong classification is the optimal Bayes rule, that is, it assign a unit to the population it has the largest posterior probability of coming from

$$R_1 = \left\{ x : \frac{f_1(x_0)}{f_2(x_0)} > \frac{\pi_2}{\pi_1} \right\}$$
$$R_2 = \left\{ x : \frac{f_1(x_0)}{f_2(x_0)} < \frac{\pi_2}{\pi_1} \right\}$$

the rule that we will use most of the cases is this rule.
Now the problem is still we said nothing on how to obtain the multivariate likelihoods (f_1 and f_2).
One way to do it is to define a model for $f_1(x)$ and $f_2(x)$ and then need to estimate the parameters. If $f_1(x)$ and $f_2(x)$ are multivariate normal distributions we have that the densities are

$$f_1(x) = (2\pi)^{-p/2} \det {\Sigma_1}^{-1/2} \exp\left\{ -\frac{1}{2}(x-\mu_1)^T \Sigma_1^{-1}(x-\mu_1) \right\}$$
$$f_2(x) = (2\pi)^{-p/2} \det {\Sigma_2}^{-1/2} \exp\left\{ -\frac{1}{2}(x-\mu_2)^T \Sigma_2^{-1}(x-\mu_2) \right\}$$

Then the likelihoods ratio becomes

$$\frac{f_1(x)}{f_2(x)} = \ldots = \det {\Sigma_1}^{-1/2} \det {\Sigma_2}^{-1/2} \exp\left\{ -\frac{1}{2}\left[x^T(\Sigma_1 - \Sigma_2)x - 2x^T(\Sigma_1^{-1}\mu_1 - \Sigma_2^{-1}\mu_2) + \mu_1^T \Sigma_1^{-1}\mu_1 - \mu_2 \Sigma_2^{-1}\mu_2 \right] \right\}$$

People usually log this above, obtaining

$$\log\left(\frac{f_1(x)}{f_2(x)}\right) = \frac{1}{2}\log\left(\frac{\det \Sigma_2}{\det \Sigma_1}\right) - \frac{1}{2}\left[x^T(\Sigma_1^{-1} - \Sigma_2^{-1})x - 2x^T(\Sigma_1^{-1}\mu_1 - \Sigma_2^{-1}\mu_2) + \mu_1^T\Sigma_1^{-1}\mu_1 - \mu_2^T\Sigma_2^{-1}\mu_2 \right]$$
$$= Q(x)$$

$Q(x)$ is so called quadratic discriminant, a function for heteroschedastic normal population; with two mnv with different variances, the optimal surface that separate them is a quadratic surface (a parabola).
We need to compare $Q(x)$ with

TODO: CHECK

- 0 (log of 1 where the two f are equivalent)
- $log(\pi_2/\pi_1)$ if using populations with different prior

What if in case of Homoscedasticity $\Sigma_1 = \Sigma_2 = \Sigma$? in case the ratio simplifies

$$\frac{f_1(x)}{f_2(x)} = \det \Sigma^{-1/2} \det \Sigma^{1/2} \cdot \exp\left[-\frac{1}{2}x^T(\Sigma - \Sigma)x - 2x^T(\Sigma\mu_1 - \Sigma\mu_2) + \mu_1^T\Sigma\mu_1 - \mu_2^T\Sigma\mu_2 \right]$$
$$\exp\left[x^T\Sigma(\mu_1 - \mu_2) - \frac{1}{2}(\mu_1 - \mu_2)^T\Sigma(\mu_1 - \mu_2) \right]$$

Taking the log we obtain

$$\log\left(\frac{f_1(x)}{f_2(x)}\right) = x^T\Sigma^{-1}(\mu_1 - \mu_2) - \frac{1}{2}(\mu_1 - \mu_2)^T\Sigma^{-1}(\mu_1 - \mu_2)$$
$$\overset{(1)}{=} (\mu_1 - \mu_2)^T\Sigma^{-1}x - \frac{1}{2}(\mu_1 - \mu_2)^T\Sigma^{-1}(\mu_1 - \mu_2)$$

where in (1) is a scalar so i can transpose it obtaining the same. The final equation has to be compared to 0 or $log(\pi_2/\pi_1)$ as usual.
This is at the population level. At the sample level we estimate μ_1 with $\overline{x}_1$ and μ_2 with $\overline{x}_2$ and Σ with W. thus we obtain

$$(\overline{x}_1 - \overline{x}_2)^T W^{-1} x > (\overline{x}_1 - \overline{x}_2)^T W^{-1} (\overline{x}_1 - \overline{x}_2)$$

Which is called **Fisher's linear discriminant rule**
In conclusion: in the equal priors case for normal omoschedastic populations, fishers rule is optimal bayes rule.
We've obtained fisher rule without any assumption and that if the population is optimal

TODO: CHECK

Other than hypotizing a mvn distribution of X the f density can be estimated in many different way: a simple way is to use naive bayes.

5.4 Lab

we seen different dataset where we know the groups, see if discriminant function we find

5.4.1 Jobs

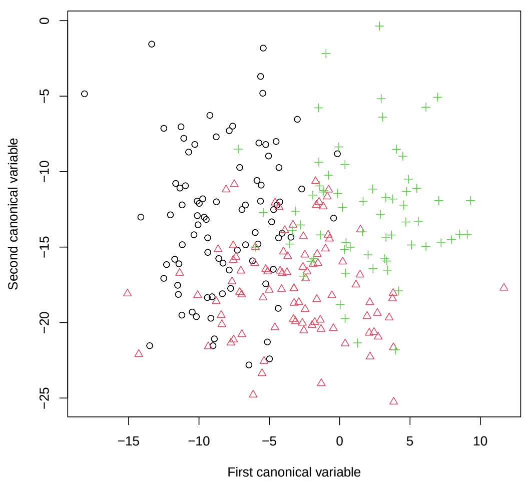
```
jobs <- read.csv("data/jobs.csv", header = TRUE, sep = ";")
head(jobs)

##   outdoor social conservative job
## 1      10     22            5   1
## 2      14     17            6   1
## 3      19     33            7   1
## 4      14     29           12   1
## 5      14     25            7   1
## 6      20     25           12   1

## data about employees belonging to (variable job)
## 1 - customer satisfaction
## 2 - mechanics
## 3 - dispatcher

unique(jobs$job)

## [1] 1 2 3

## this job classification are linked to differet personality type? (three
## variables outdoor, social and conservative activities)

jobs$job <- factor(jobs$job,
                   levels = 1:3,
                   labels = c("Customer service",
                              "Mechanic",
                              "Dispatcher"))
```

we'd like to find discriminant function: maximum number is equal to minimum betweeb $G-1$ and P. here $G=3$ and $p=3$. maximum number of canonical variable we can have is 2. we'd like to find this variables

```
## we
p <- ncol(jobs) - 1
G <- length(unique(jobs$job))
n_tot <- nrow(jobs)   # total number of observation in all the two groups


## we build the sub-dataset: dataset with only emplyees with each type
jobs_split <- split(jobs[, -4], jobs$job)  # we don't want the classification variables here fo
lapply(jobs_split, head)

## $`Customer service`
##   outdoor social conservative
## 1      10     22            5
## 2      14     17            6
## 3      19     33            7
## 4      14     29           12
## 5      14     25            7
## 6      20     25           12
##
## $Mechanic
##    outdoor social conservative
## 86      20     27            6
## 87      21     15           10
## 88      15     27           12
## 89      15     29            8
## 90      11     25           11
## 91      24      9           17
##
## $Dispatcher
##     outdoor social conservative
## 179      19     19           16
## 180      17     17           12
## 181       8     17           14
## 182      13     20           16
## 183      14     18            4
## 184      17     12           13

# now we want to extract the number of observation in each group
n_g <- sapply(jobs_split, nrow) # to ease, have it as a vector

## now we have all we need to perform LDA. we need to compute within and
## between group covariance matrix and then W^{-1}B - ...

## within group covariance matrix: we need just Sg (covaraince matrix of each group)

(Sg <- lapply(jobs_split, var)) # variance covariance matrix
```

```
## $`Customer service`
##                 outdoor    social conservative
## outdoor      21.609804  4.418627    -2.548039
## social        4.418627 18.794678     2.506583
## conservative -2.548039  2.506583     9.880392
##
## $Mechanic
##                 outdoor   social conservative
## outdoor      12.707807 -1.57597     2.543595
## social       -1.575970 20.70851     0.175900
## conservative  2.543595  0.17590    10.512856
##
## $Dispatcher
##                  outdoor     social conservative
## outdoor      16.8941725  1.6727273    0.6890443
## social        1.6727273 14.1902098    0.1342657
## conservative  0.6890443  0.1342657   13.6326340

## compute W = 1/n-G sum_g (n_g - 1) Sg

## build the object where to store the numbers

W <- matrix(0, nrow = p, ncol = p) # covariance il p x p
for (g in 1:G){ # loop for each group
  W <- W + (n_g[g] - 1) * Sg[[g]] # in Sg a double square brackets because a list

}

## we need to divide W for n-G
W <- W/(n_tot - G) # final within group covariance matrix


## Between group covariance matrix: the method is the same with for loop

xbarg <- lapply(jobs_split, colMeans) # average of variable in g-th group
xbar <- colMeans(jobs[, -4]) # overall average

B <- matrix(0, nrow = p, ncol = p) # covariance il p x p
for (g in 1:G){ # loop for each group
  B <- B + n_g[g] * (xbarg[[g]]  - xbar) %*% t(xbarg[[g]]  - xbar)
}

(B <- B/(G-1))# final between group covariance matrix

##         outdoor    social conservative
## [1,]  804.8996 -397.1973     141.5855
## [2,] -397.1973 1444.5614    -702.9200
## [3,]  141.5855 -702.9200     345.8797

## now we do SVD of W^{-1} B: we use eigen, SVD of (W^-1 B)
```

```
spectral <- eigen(solve(W) %*% B)

round(spectral$values,2) # only the first two are different from zero: this was

## [1] 130.20  38.62   0.00

# expected: the maximum number of discriminant functions was 2

## now we extract the canonical variables: the eigenvectors corresponding to
## not null eigenvalue

(A <- spectral$vectors[, 1:2])

##            [,1]       [,2]
## [1,]  0.3470958 -0.9130164
## [2,] -0.7331079 -0.2019912
## [3,]  0.5848738  0.3544018

colnames(jobs[, -4])

## [1] "outdoor"      "social"       "conservative"

## A is 3x2 with rows we have outdoor social and conservative, and in column
## thefunction obtained
## trying to interpret: look at the coefficiente: the first function has a
## positive value for first and third, while negative for social. It separates
## according to the social activities
## the second function has high negative value for outdoor, then a less
## negative for social and finaly a positive for conservative activities: seems
## a linear constrast of outtoor vs conservative activity

## similarly to PC we can find linear comb between our data and the
## discriminant function (project original data)

## we need just to transform our data in matrix

X <- as.matrix(jobs[, -4])
Y <- X %*% A # linear combination

plot(Y,
     xlab = "First canonical variable",
     ylab = "Second canonical variable",
     col = as.integer(jobs$job),
     pch = as.integer(jobs$job))
```



```
## these three groups seems not to be very separated according to first two
## canonical variables

## now we check our classification: assume we need to assign a new observation
## (of unknown)
## compare the distance of each observation to the average of the group, and
## then assign the observation to the group it is closer to. we can use the
## euclidean distance on the derived subspace or mahalanobis on the original
## subspace

## we need to project the average of the group in the derived/discriminant
## subspaces
## that is we need to have y_g

ybarg <- xbarg

for (g in 1:G){
  ybarg[[g]] <- xbarg[[g]] %*%  A
}

## euclidean distance of every observation from the ybarg and assign the
## observation to the classwith the smallest

## we need to compute euclidean distance for every group

## we build a martrix empty at the beginning

# for each row the observation and in each column the group
# in the last column we put the assigned group (1, 2 or 3)
euclideanD <- matrix(NA, nrow = n_tot, ncol = G+1) # in 1,2 the distance between firs
```

```
for (g in 1:G){
  # the g-th column contains the distance of every row with the g-th group
  euclideanD[, g] <- apply(Y, 1, function(y){sqrt(sum((y - ybarg[[g]])^2))})

}

head(euclideanD)
```

```
##          [,1]      [,2]      [,3] [,4]
## [1,] 2.073109  8.786252 11.585664   NA
## [2,] 4.155567  3.641184  6.088613   NA
## [3,] 9.975598 11.090576 17.713411   NA
## [4,] 1.774625  7.027652 11.337834   NA
## [5,] 2.548430  6.634660 11.517190   NA
## [6,] 7.029186  1.908548  8.908554   NA
```

```
euclideanD[, G+1] <- apply(euclideanD[, 1:G], 1, which.min)
head(euclideanD) # the last columns has the assigment
```

```
##          [,1]      [,2]      [,3] [,4]
## [1,] 2.073109  8.786252 11.585664    1
## [2,] 4.155567  3.641184  6.088613    2
## [3,] 9.975598 11.090576 17.713411    1
## [4,] 1.774625  7.027652 11.337834    1
## [5,] 2.548430  6.634660 11.517190    1
## [6,] 7.029186  1.908548  8.908554    2
```

```
## now we can do the same with malahanobis distance but on the original
## variable
## structure is the same of the euclidean, we work on X instead

mahalanobisD <- matrix(NA, nrow = n_tot, ncol = G+1)
for (g in 1:G){
  # the g-th column contains the distance of every row with the g-th group
  # we work with X instead of Y
  mahalanobisD[, g] <- apply(X, 1, function(x) {
    t(x - xbarg[[g]]) %*% solve(W) %*% (x - xbarg[[g]])  # check
  })
}
mahalanobisD[, G+1] <- apply(mahalanobisD[, 1:G], 1, which.min)
head(mahalanobisD) # the last columns has the assigment
```

```
##           [,1]     [,2]      [,3] [,4]
## [1,] 1.9551699 6.772098 11.089305    1
## [2,] 3.7381503 3.348347  5.117959    2
## [3,] 6.8131739 8.986827 21.781321    1
## [4,] 1.9842879 5.148524 10.776851    1
## [5,] 0.5443612 3.178535  9.381197    1
## [6,] 4.0391721 1.139074  6.087088    2
```

```
## elements on the main diagonal indicates units correctly classified
table(jobs$job, mahalanobisD[, 4])
```

```
##
##                      1  2  3
##   Customer service 70 11  4
##   Mechanic         16 62 15
##   Dispatcher        3 12 51

## missclassification rates
mean(as.integer(jobs$job) != mahalanobisD[, 4])

## [1] 0.25

## 25% of obs is wrongly classified
```

Until here all done manually. to perform the analysis using function

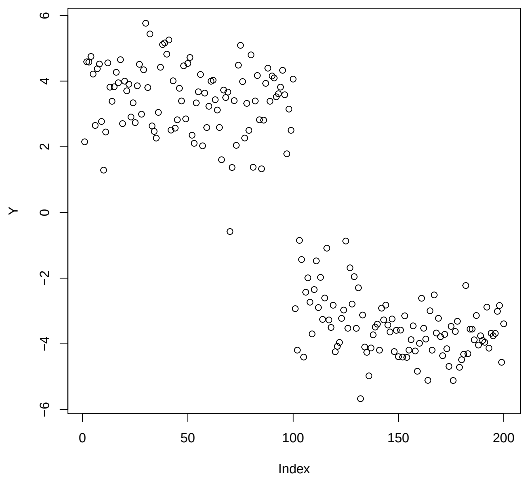
```
lda_out <- MASS::lda(x = jobs[, -4],        #X
                     grouping = jobs$job)   #y

lda_out

## Call:
## lda(jobs[, -4], grouping = jobs$job)
##
## Prior probabilities of groups:
## Customer service         Mechanic       Dispatcher
##        0.3483607        0.3811475        0.2704918
##
## Group means:
##                   outdoor   social conservative
## Customer service 12.51765 24.22353     9.023529
## Mechanic         18.53763 21.13978    10.139785
## Dispatcher       15.57576 15.45455    13.242424
##
## Coefficients of linear discriminants:
##                      LD1         LD2
## outdoor       0.09198065 -0.22501431
## social       -0.19427415 -0.04978105
## conservative  0.15499199  0.08734288
##
## Proportion of trace:
##    LD1    LD2
## 0.7712 0.2288

## coefficients of linear discriminants is what we called A.
## discrminant function are equivalent up to sign changes

A

##              [,1]       [,2]
## [1,]  0.3470958 -0.9130164
## [2,] -0.7331079 -0.2019912
## [3,]  0.5848738  0.3544018

## but results are different: lda uses a different constraint put on the within
## group covariance matrix, set to spherical

## usually we have this
##
## A^T W A = Psi
##
## what lda does is to consider A^* such that
## t(A^*)W A^* = I

## if we want to obtain the same resilts as lda we need to compute this and we
## get that
## A^* = A Psi^{-1/2}

## if we want to check

Psi = t(A) %*% W %*% A  #diagonal matrix (out of are almost zero)
## to have perfect diag
Psi = diag(diag(Psi))

Astar = A %*% solve(Psi^{1/2})
Astar

##             [,1]        [,2]
## [1,]  0.09198065 -0.22501431
## [2,] -0.19427415 -0.04978105
## [3,]  0.15499199  0.08734288

lda_out$scaling # same

##                      LD1         LD2
## outdoor       0.09198065 -0.22501431
## social       -0.19427415 -0.04978105
## conservative  0.15499199  0.08734288

## if we use Astar instead of A we get identity
t(Astar) %*% W %*% Astar

##               [,1]          [,2]
## [1,]  1.000000e+00 -6.106227e-16
## [2,] -6.938894e-16  1.000000e+00

## last thing we do with lda:
## lda does not report the projection on the unidimensional space, we use the
## predict function

pred <- predict(lda_out) # class, prediction ,posterior is the posterior prob
# matrix and x is the projected units
```

```
Ylda <- pred$x
head(Ylda) # hould beequal

##             LD1         LD2
## [1,] -1.6423155  0.71477348
## [2,] -0.1480302  0.15096436
## [3,] -2.6415213 -1.68326115
## [4,] -1.5493681  0.07764901
## [5,] -1.5472314 -0.15994117
## [6,] -0.2203876 -1.07331266

head(Y) # bvery differetn

##             [,1]      [,2]
## [1,]  -9.733047 -11.80196
## [2,]  -4.094251 -14.08967
## [3,] -13.503624 -21.53221
## [4,]  -9.382303 -14.38715
## [5,]  -9.374240 -15.35120
## [6,]  -4.367296 -19.05729

## different because use Astar instad of A.
## also if we try to use Astar does not work because data is centered by lda

# so we need to center X to obtain the same results
head(scale(X, center = TRUE, scale = FALSE) %*% Astar)

##             [,1]        [,2]
## [1,] -1.6423155  0.71477348
## [2,] -0.1480302  0.15096436
## [3,] -2.6415213 -1.68326115
## [4,] -1.5493681  0.07764901
## [5,] -1.5472314 -0.15994117
## [6,] -0.2203876 -1.07331266

# in this way we have the same results
```

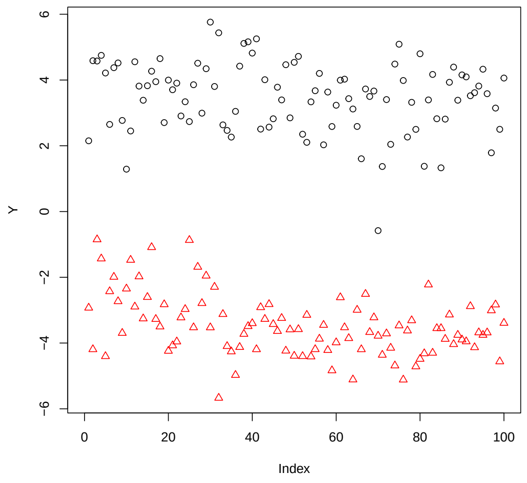
5.4.2 banknotes

Two dataset with `lda`

```
head(banknotes <- read.csv("data/banknotes.csv", sep = ";"))

##   Length  Left Right Bottom  Top Diagonal
## 1  214.8 131.0 131.1    9.0  9.7    141.0
## 2  214.6 129.7 129.7    8.1  9.5    141.7
## 3  214.8 129.7 129.7    8.7  9.6    142.2
## 4  214.8 129.7 129.6    7.5 10.4    142.0
## 5  215.0 129.6 129.7   10.4  7.7    141.8
## 6  215.7 130.8 130.5    9.0 10.1    141.4
## contain info on 200 banknotes
## first 100 are true/genuine
## last 100 are false
group <- c(rep("Genuine", 100), rep("Counterfeit", 100))

## we see if with data the two group are well separated
## min of G-1=1 and p=6 is 1.
## we're able to compute only 1 discriminant function

(lda_out <- MASS::lda(x=banknotes, grouping = group))

## Call:
## lda(banknotes, grouping = group)
##
## Prior probabilities of groups:
## Counterfeit     Genuine
##         0.5         0.5
##
## Group means:
##              Length    Left   Right Bottom    Top Diagonal
## Counterfeit 214.823 130.300 130.193 10.530 11.133  139.450
## Genuine     214.969 129.943 129.720  8.305 10.168  141.517
##
## Coefficients of linear discriminants:
##                   LD1
## Length    0.005011113
## Left      0.832432523
## Right    -0.848993093
## Bottom   -1.117335597
## Top      -1.178884468
## Diagonal  1.556520967

# we see only 1 discriminant function has been computed

## we have positive value for left and diagonal and negative for other

## check the missclassification rate
pred <- predict(lda_out)

class_notes <- pred$class
table(class_notes, group) # almost perfect (only 1 banknotes misclassified)

##              group
## class_notes   Counterfeit Genuine
##   Counterfeit         100       1
##   Genuine               0      99

## to see the plot on the unique dimension (on x we put just an index)
Y <- pred$x
plot(Y)
```



```
plot(Y, type = "n", xlim = c(0, 100)) # don't plot anithg but set in 0:100
points(Y[1:100], col = 'black') # true banknotes
points(Y[101:200], col = 'red', pch = 2) # true banknotes
```



```
# this way we ignore the inde
```

5.4.3   wdbc

```
if (FALSE) {

  wdbc <- read.csv(url("https://archive.ics.uci.edu/ml/machine-learning-databases/bre
                  header = FALSE,
                  na.strings = "?")


## UCI machine learning repo: first is M(malignant) or B (bening) and  30 regressors of cancer
## malignant or benign

# remove the first column (id of the  entry)
wdbc = wdbc[, -1]
names(wdbc)[1] = "response"

write.csv(x = wdbc, file = "multivariate_statistics/data/wdbc.csv", row.names = FALSE)
}

wdbc <- read.csv("data/wdbc.csv")


## onlyu diffirent thing is subdivision in training and test set: until now
## missclassification rate on the same data that

## divide the dataset using sample
set.seed(19)
n <- nrow(wdbc)
train <- sample(n, size = n * 2/3, replace =FALSE) # 2/3 to train  1/3 to test

wdbc_train <- wdbc[train, ]
wdbc_test <- wdbc[-train, ]

lda_out <- MASS::lda(x = wdbc_train[, -1], grouping = wdbc_train[, 1])
class_train <- predict(lda_out)$class

## wrong missglassification er
1 - sum(diag(table(wdbc_train$response, class_train))) /
  sum((table(wdbc_train$response, class_train))) # very low

## [1] 0.02902375

# actulamissclassification
class_test <- predict(lda_out, newdata = wdbc_test[, -1])$class
1 - sum(diag(table(wdbc_test$response, class_test))) /
  sum((table(wdbc_test$response, class_test))) # very low

## [1] 0.07894737

# error higher compared to previous
```