Prelims Statistics and Data Analysis
Lectures 11-16

Frank Windmeijer
frank.windmeijer@stats.ox.ac.uk

Trinity Term 2026, Version of May 22, 2026

1. Introduction and Motivating Examples
2. Data Visualisation
3. The Multivariate Normal Distribution
4. Principal Components Analysis
5. Clustering

Statistical learning

So far, this course introduced parameter estimation in statistical models: maximum likelihood, confidence intervals, and linear regression. The rest of the course is an introduction to **statistical learning** framework and **unsupervised learning** in particular.

Statistical learning refers to a vast set of tools for understanding (typically large quantities of) data, and is closely related to **Machine Learning**, **Data Science** and **Artificial Intelligence**.

Examples of recent advances in AI which make use of **machine learning** models:
learning game strategies from sensory input, computer vision, machine translation, AlphaGO, ChatGPT.

Statistical learning

Massive amounts of data are being collected in many different fields.

Financial institutions, businesses, governments, hospitals, and universities are all interested in utilizing and making sense of data they collect.

The majority of mathematics students will go on to work in careers that involve carrying out or interpreting analysis of data.

This course leads onto several more advanced courses offered by the Department of Statistics, including *Part B Statistical Machine Learning* and *Part C Advanced Topics in Statistical Machine Learning*.

Supervised vs unsupervised learning

$$Y = \beta_0 + \sum_{j=1}^{p} \beta_j x_j + \epsilon, \qquad \epsilon \sim N(0, \sigma^2)$$

Linear regression is an example of **supervised learning**.

i.e. we build a model to predict a response variable Y using a set of p variables (or features) $x_1, \ldots, x_p$.

Typically, we will have data on n observations.

Some references to papers with clustering methods:

http://dx.doi.org/10.3390/electronics9081295

Supervised vs unsupervised learning

In **unsupervised learning** we just have observations on the set of variables $X_1, \ldots, X_p$, measured on n observations.

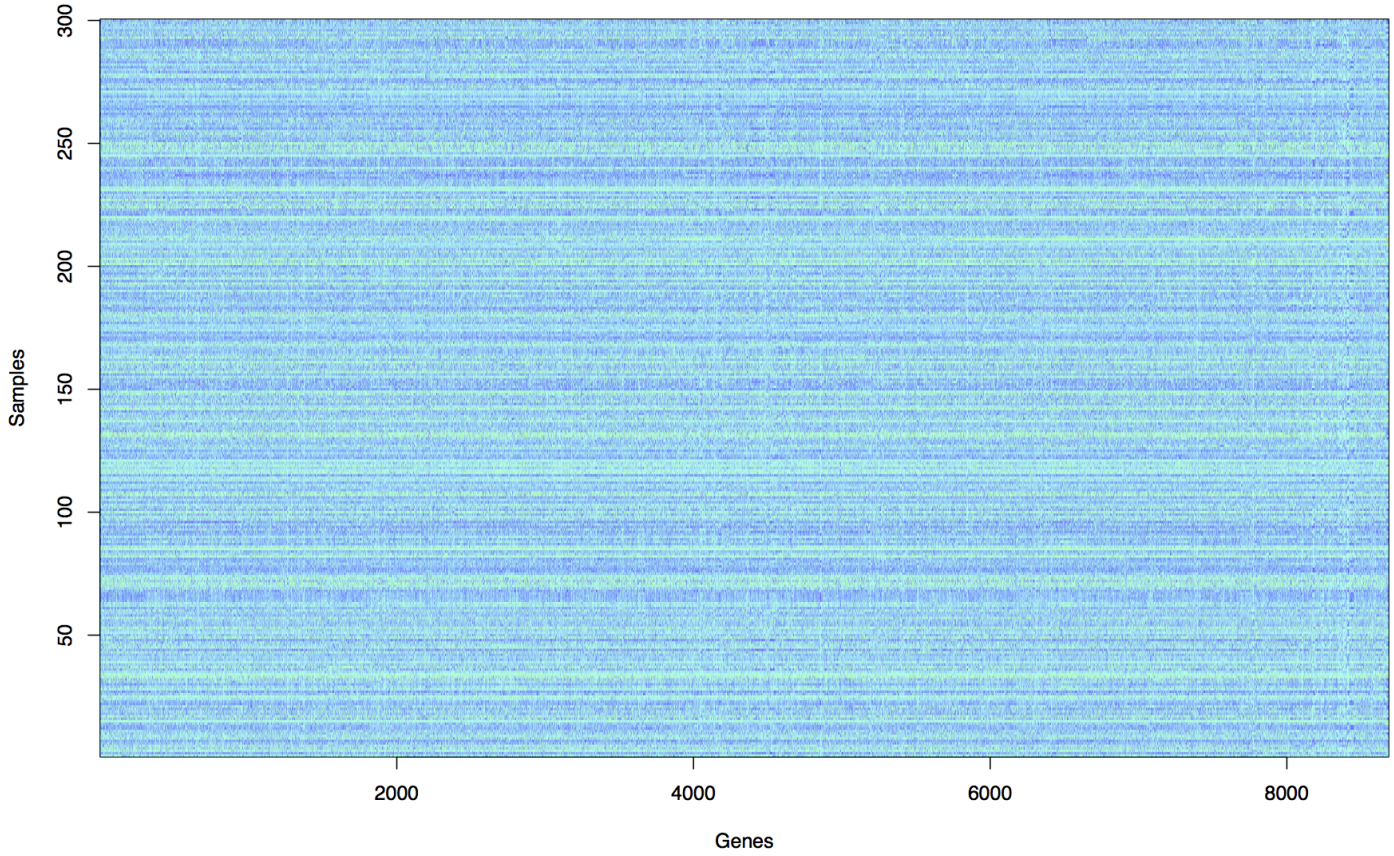
Interest lies in looking for patterns and structure in the data, which is often large and high-dimensional.

Relevant questions include

1. Can we find a way to visualize the data that is informative?
2. Can we compress the dataset without losing any relevant information?
3. Can we find separate subgroups (or clusters) of observations that describe the structure of the dataset?

Motivating example 1

300 cells each with measurements of activity of 8,686 genes (visit movie.gif here `https://courses.maths.ox.ac.uk/course/view.php?id=6033` for 3D PCA projection)



Motivating example 2

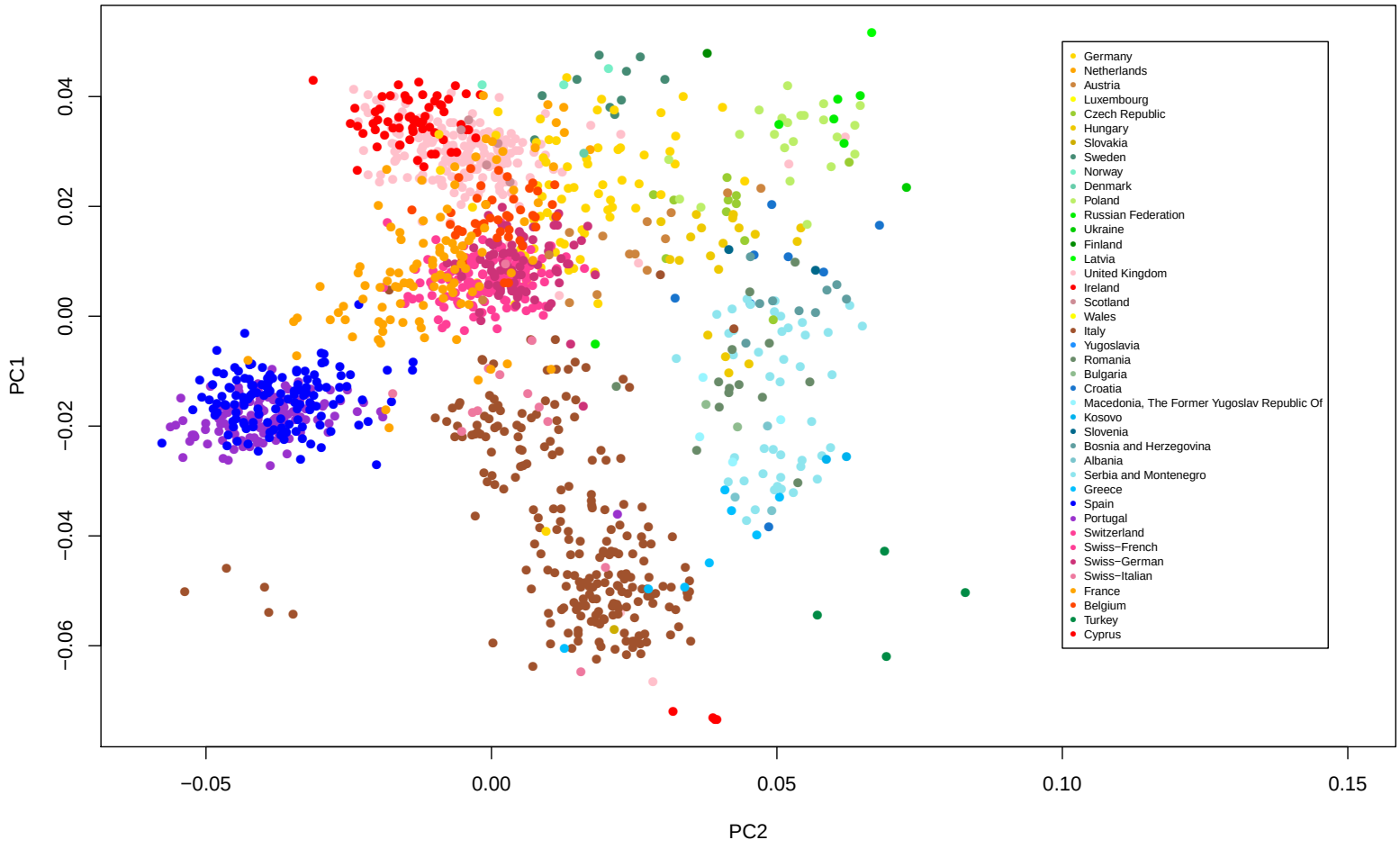
3,000 individuals from different European countries, each with measurements at ~500,000 genes.

From the paper by Novembre et al. (2008) *Nature* 456:98-101

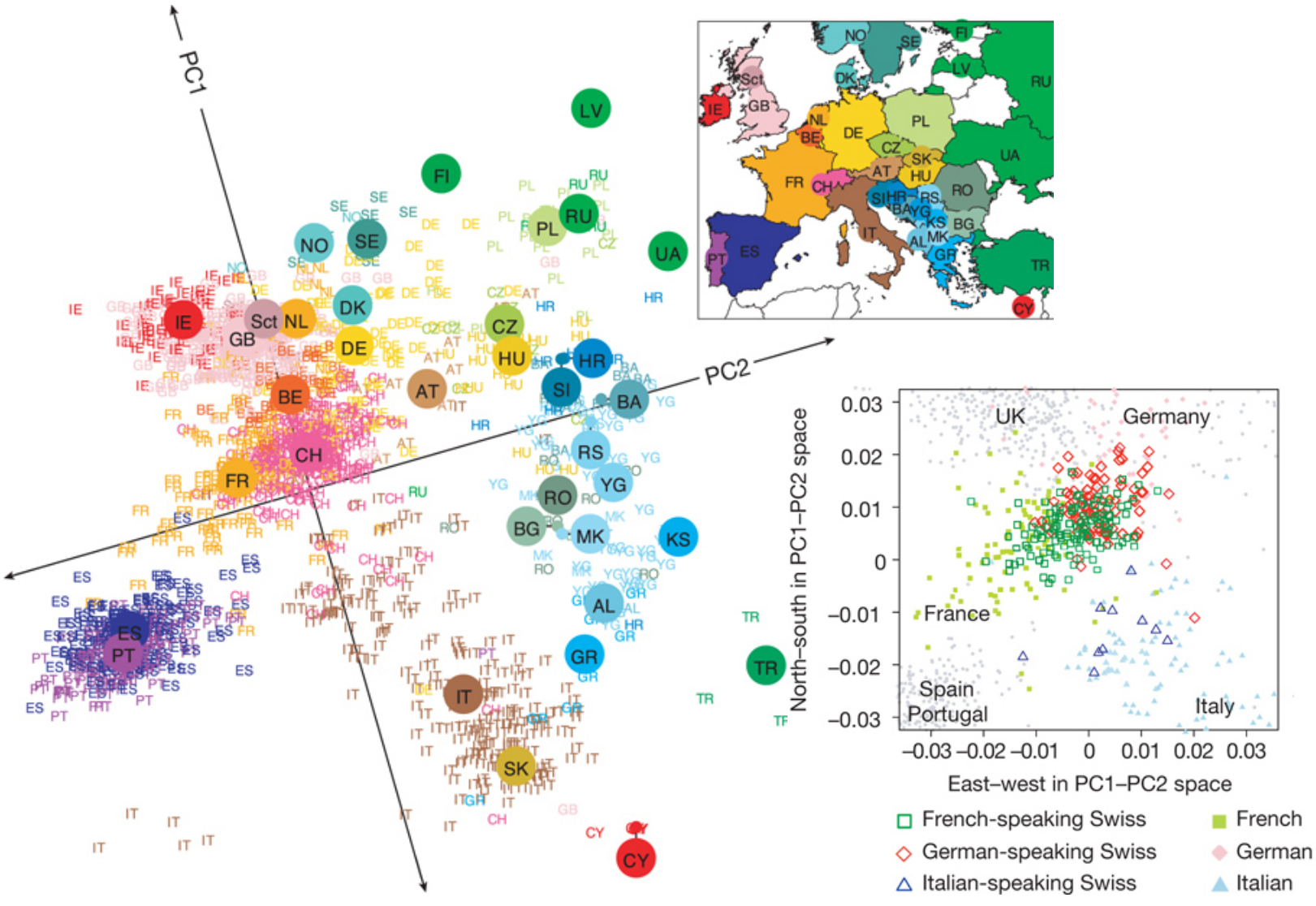
Scientific question

"not clear to what extent populations within continental regions exist as discrete genetic clusters versus as a genetic continuum, nor how precisely one can assign an individual to a geographic location on the basis of their genetic information alone."

Motivating example 2



Motivating example 2



Genes mirror geography within Europe, Nature 2008

Data matrices and notation - I

A dataset that consists of n observations of p variables.
We represent the data in an $n \times p$ matrix $\boldsymbol{X}$, where x_{ij} is the observed value of the jth variable for the ith observation.

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix}.$$

We will refer to $\boldsymbol{X}$ as the **data matrix**.

Data matrices and notation - II

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix}.$$

The observations are rows. The variables are columns.
The rows of $\boldsymbol{X}$ are $x_1^T, x_2^T, \dots, x_n^T$.

Data matrices and notation - III

The rows of $\boldsymbol{X}$ are $x_1^T, x_2^T, \ldots, x_n^T$.
Here x_i is a vector of length p, containing the p variable measurements for the ith observation. That is

$$x_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{pmatrix} = (x_{i1}, \ldots, x_{ip})^T$$

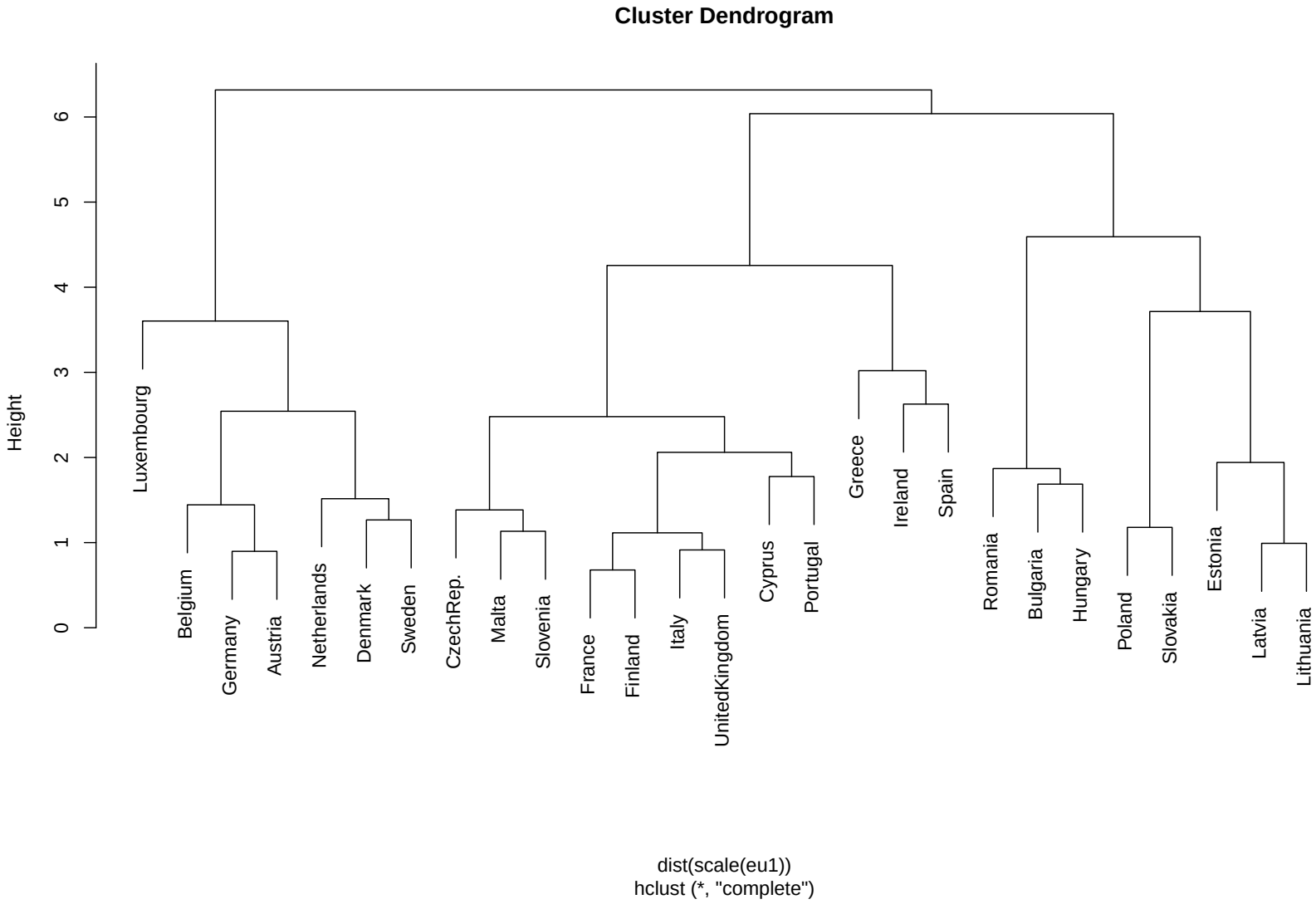
Vectors treated as column vectors by default.

Motivating example 3

Economic indicators for 27 EU countries (data from 2012)

Country	CPI	UNE	INP	BOP	PRC	UN%
Belgium	116.03	4.77	125.59	908.60	6716.50	-1.60
Bulgaria	141.20	7.31	102.39	27.80	1094.70	3.50
CzechRep.	116.20	4.88	119.01	-277.90	2616.40	-0.60
Denmark	114.20	6.03	88.20	1156.40	7992.40	0.50
Germany	111.60	4.63	111.30	499.40	6774.60	-1.30
Estonia	135.08	9.71	111.50	153.40	2194.10	-7.70
Ireland	106.80	10.20	111.20	-166.50	6525.10	2.00
Greece	122.83	11.30	78.22	-764.10	5620.10	6.40
Spain	116.97	15.79	83.44	-280.80	4955.80	0.70
France	111.55	6.77	92.60	-337.10	6828.50	-0.90
Italy	115.00	5.05	87.80	-366.20	5996.60	-0.50
Cyprus	116.44	5.14	86.91	-1090.60	5310.30	-0.40
Latvia	144.47	12.11	110.39	42.30	1968.30	-3.60
Lithuania	135.08	11.47	114.50	-77.40	2130.60	-4.30
Luxembourg	118.19	3.14	85.51	2016.50	10051.60	-3.00
Hungary	134.66	6.77	115.10	156.20	1954.80	-0.10
Malta	117.65	4.15	101.65	359.40	3378.30	-0.60
Netherlands	111.17	3.23	103.80	1156.60	6046.00	-0.40
Austria	114.10	2.99	116.80	87.80	7045.50	-1.50
Poland	119.90	6.28	146.70	-74.80	2124.20	-1.00
Portugal	113.06	9.68	89.30	-613.40	4073.60	0.80
Romania	142.34	4.76	131.80	-128.70	1302.20	3.20
Slovenia	118.33	5.56	105.40	39.40	3528.30	1.80
Slovakia	117.17	9.19	156.30	16.00	2515.30	-2.10
Finland	114.60	5.92	101.00	-503.70	7198.80	-1.30
Sweden	112.71	6.10	100.50	1079.10	7476.70	-2.30
UnitedKingdom	120.90	6.11	90.36	-24.30	6843.90	-0.80

Motivating example 3



Data visualisation

Campbell (1974) studied rock crabs of the genus leptograpsus. One species, L. variegatus, had been split into two new species according to their colour: orange and blue. Preserved specimens lose their colour, so it was hoped that morphological differences would enable museum material to be classified. Data are available on 50 specimens of each sex of each species.

Each specimen has measurements on:

- the width of the frontal lobe (FL),
- the rear width (RW),
- the length along the carapace midline (CL),
- the maximum width (CW) of the carapace,
- the body depth (BD) in mm.

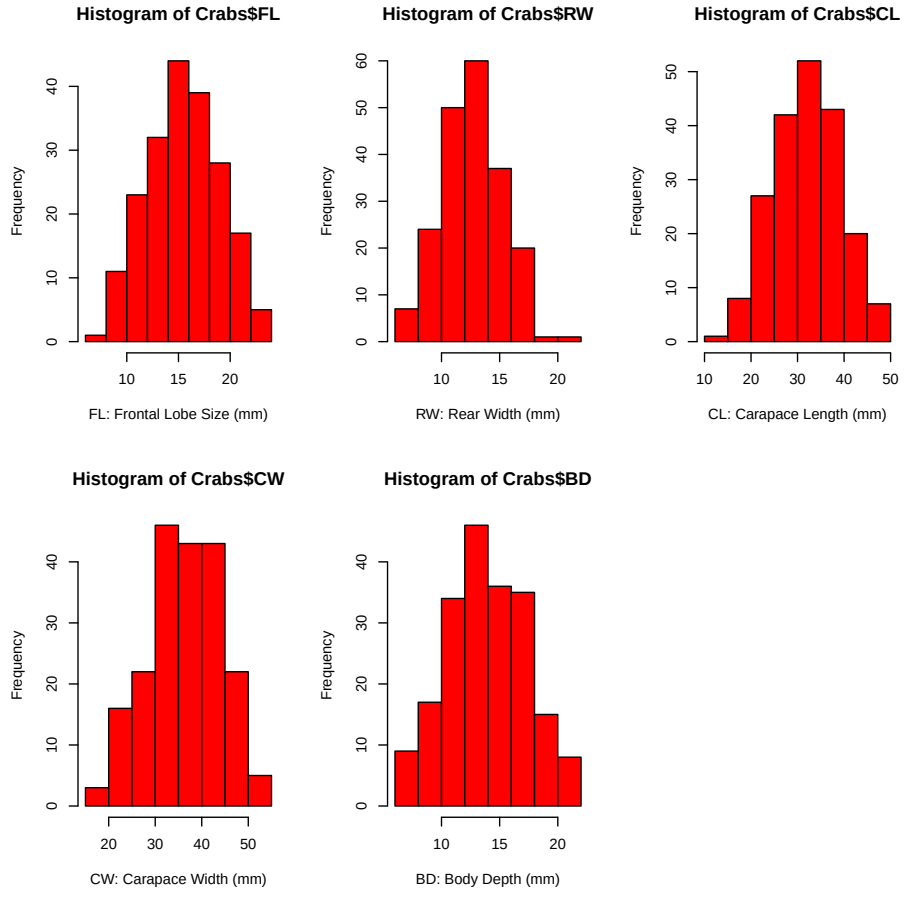
So the data matrix $\boldsymbol{X}$ has dimensions 200×5.

Crabs Data

	FL	RW	CL	CW	BD
1	8.1	6.7	16.1	19.0	7.0
2	8.8	7.7	18.1	20.8	7.4
3	9.2	7.8	19.0	22.4	7.7
4	9.6	7.9	20.1	23.1	8.2
5	9.8	8.0	20.3	23.0	8.2
6	10.8	9.0	23.0	26.5	9.8
7	11.1	9.9	23.8	27.1	9.8
8	11.6	9.1	24.5	28.4	10.4
9	11.8	9.6	24.2	27.8	9.7
10	11.8	10.5	25.2	29.3	10.3
11	12.2	10.8	27.3	31.6	10.9
12	12.3	11.0	26.8	31.5	11.4
13	12.6	10.0	27.7	31.7	11.4
14	12.8	10.2	27.2	31.8	10.9
15	12.8	10.9	27.4	31.5	11.0
16	12.9	11.0	26.8	30.9	11.4
17	13.1	10.6	28.2	32.3	11.0
18	13.1	10.9	28.3	32.4	11.2
19	13.3	11.1	27.8	32.3	11.3
20	13.9	11.1	29.2	33.3	12.1

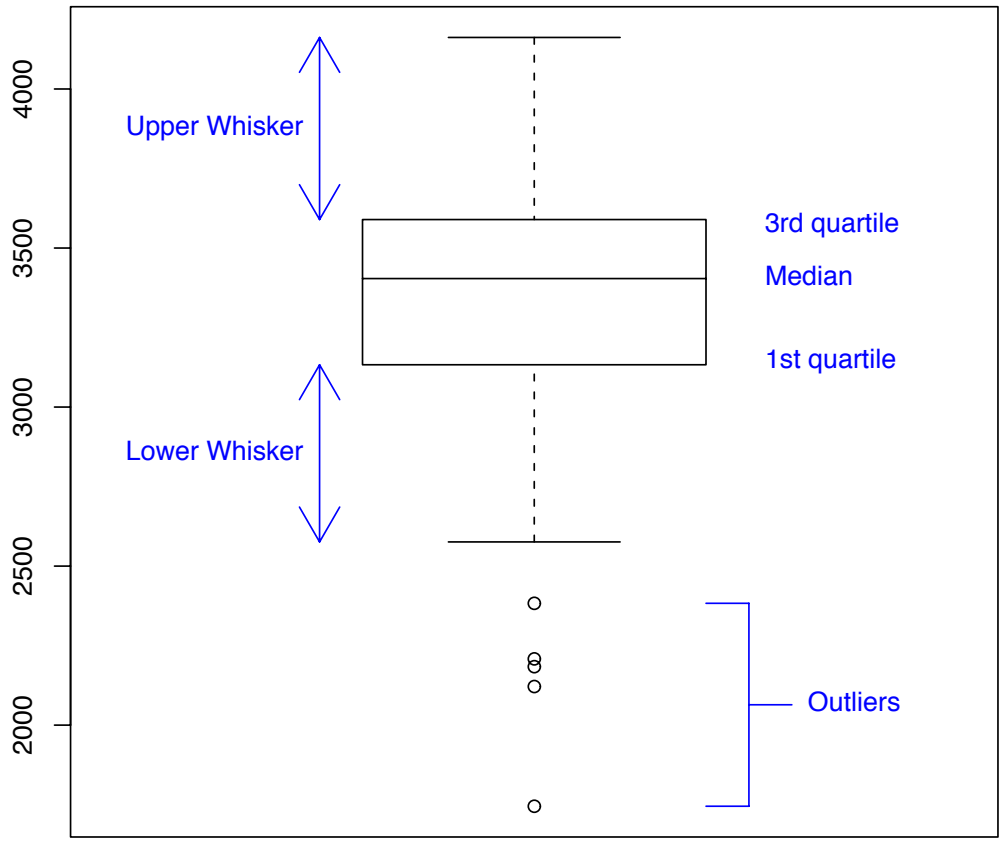
Histograms

A histogram is one of the simplest ways of visualising the data from a single variable.



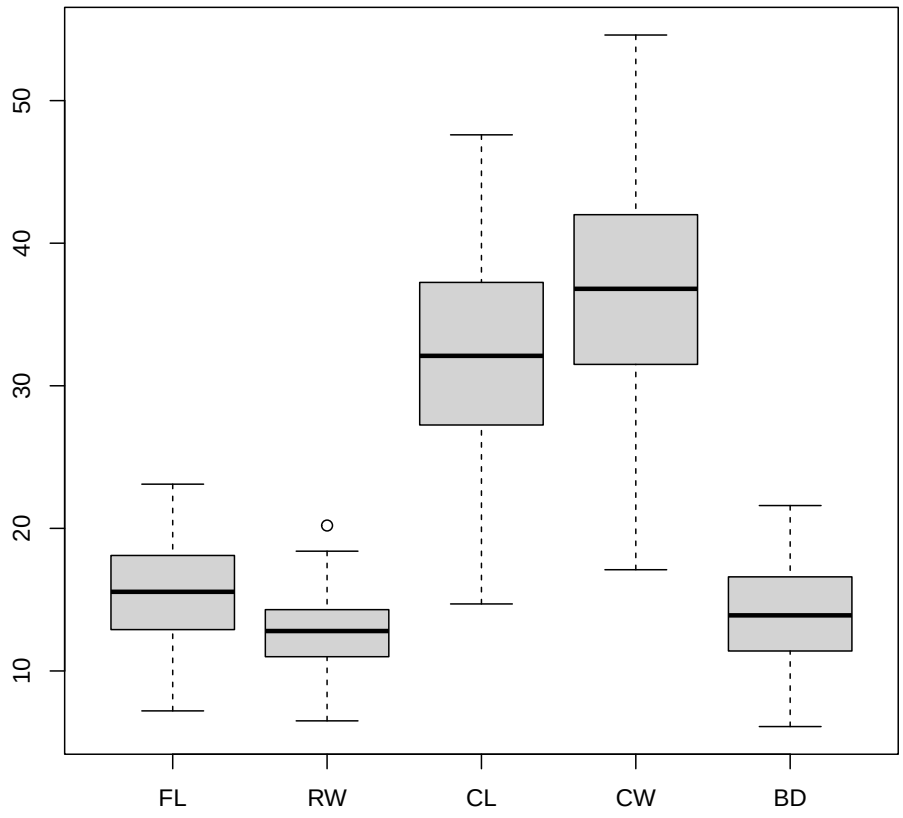
Boxplots

A Box Plot (sometimes called a Box-and-Whisker Plot) is a relatively sophisticated plot that summarises the distribution of a given variable.



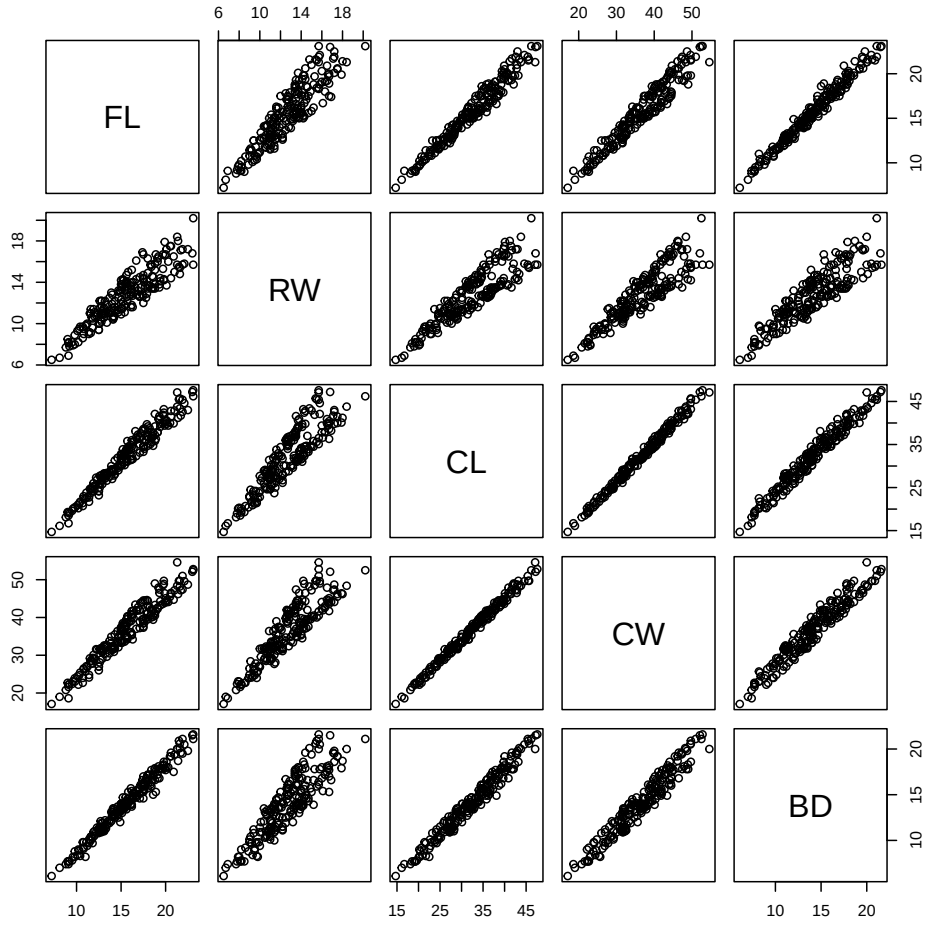
Boxplots

Boxplots of the crabs dataset



Pairs plots

Plotting pairs of variables together in a scatter plot can be helpful to see how variables co-vary.



The Multivariate Normal Distribution - I

If $X \sim N(\mu, \sigma^2)$ then the pdf of the univariate normal distribution is

$$f(x) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\right) \quad -\infty < x < \infty$$

The Multivariate Normal Distribution - II

Suppose $X = (X_1, \ldots, X_p)^T$ is a p-dimensional random column vector. X has a multivariate normal distribution (MVN) with mean p-column vector $\mu = (\mu_1, \ldots, \mu_p)^T$ and $p \times p$ covariance matrix Σ, if the joint pdf of X is

$$f(x) = \frac{1}{(2\pi)^{p/2}|\mathbf{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\mathbf{\Sigma}^{-1}(x-\mu)\right), \quad x \in \mathbb{R}^p$$

We use the notation $X \sim N_p(\mu, \mathbf{\Sigma})$. When $p = 1$ this reduces to the univariate normal distribution. The multivariate normal distribution is sometimes referred to as the multivariate Gaussian distribution.

The Multivariate Normal Distribution - III

The interpretation of the parameters is as follows

$$E(X_j) = \mu_j \qquad \text{for } j \in 1, \ldots, p$$

$$\text{var}(X_j) = \mathbf{\Sigma}_{jj} \qquad \text{for } j \in 1, \ldots, p$$

$$\text{cov}(X_j, X_k) = \mathbf{\Sigma}_{jk} \qquad \text{for } j \neq k \in 1, \ldots, p$$

The correlation between two variables X_j and X_k is defined as follow

$$\text{cor}(X_j, X_k) = \frac{\text{cov}(X_j, X_k)}{\sqrt{\text{var}(X_j)\,\text{var}(X_k)}} = \frac{\mathbf{\Sigma}_{jk}}{\sqrt{\mathbf{\Sigma}_{jj}\mathbf{\Sigma}_{kk}}} \in [-1, 1]$$

Example : the Bivariate normal distribution ($p = 2$)

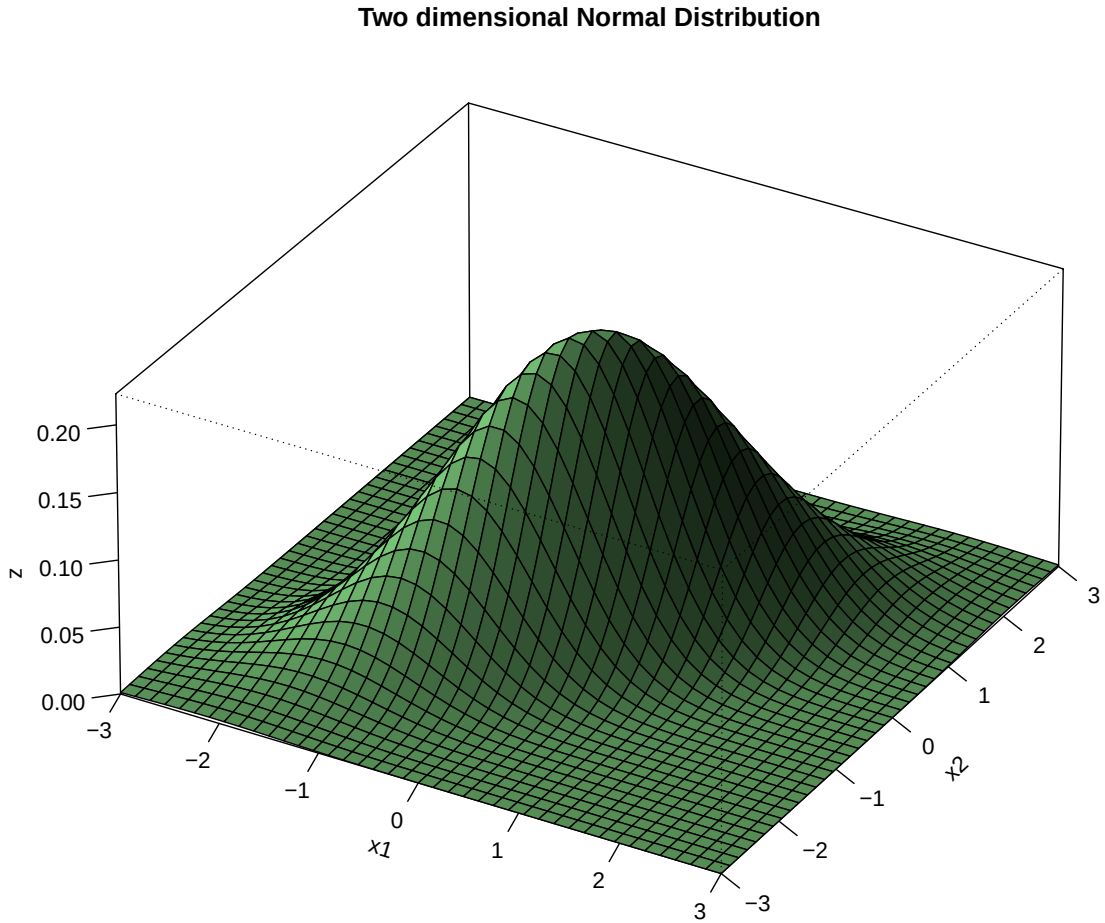
When $p = 2$ we have $X = (X_1, X_2)^T \sim N_2(\mu, \mathbf{\Sigma})$.

If $\mu = c(0,0)^T$ and $\mathbf{\Sigma} = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$ then

$$f(\boldsymbol{x}) = \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}(x_1^2 - 2\rho x_1 x_2 + x_2^2)\right)$$

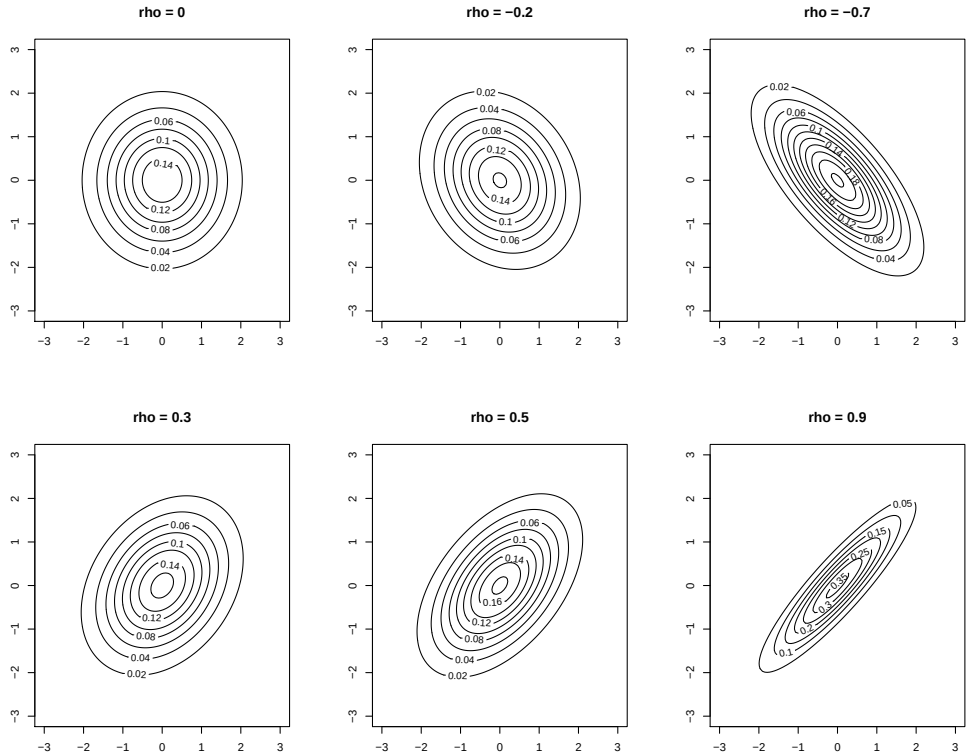
Multivariate Normal Density

$$X \sim N_2(\mu, \mathbf{\Sigma}) \text{ with } \mu = (0,0)^T \text{ and } \mathbf{\Sigma} = \begin{pmatrix} 1 & 0.7 \\ 0.7 & 1 \end{pmatrix}$$



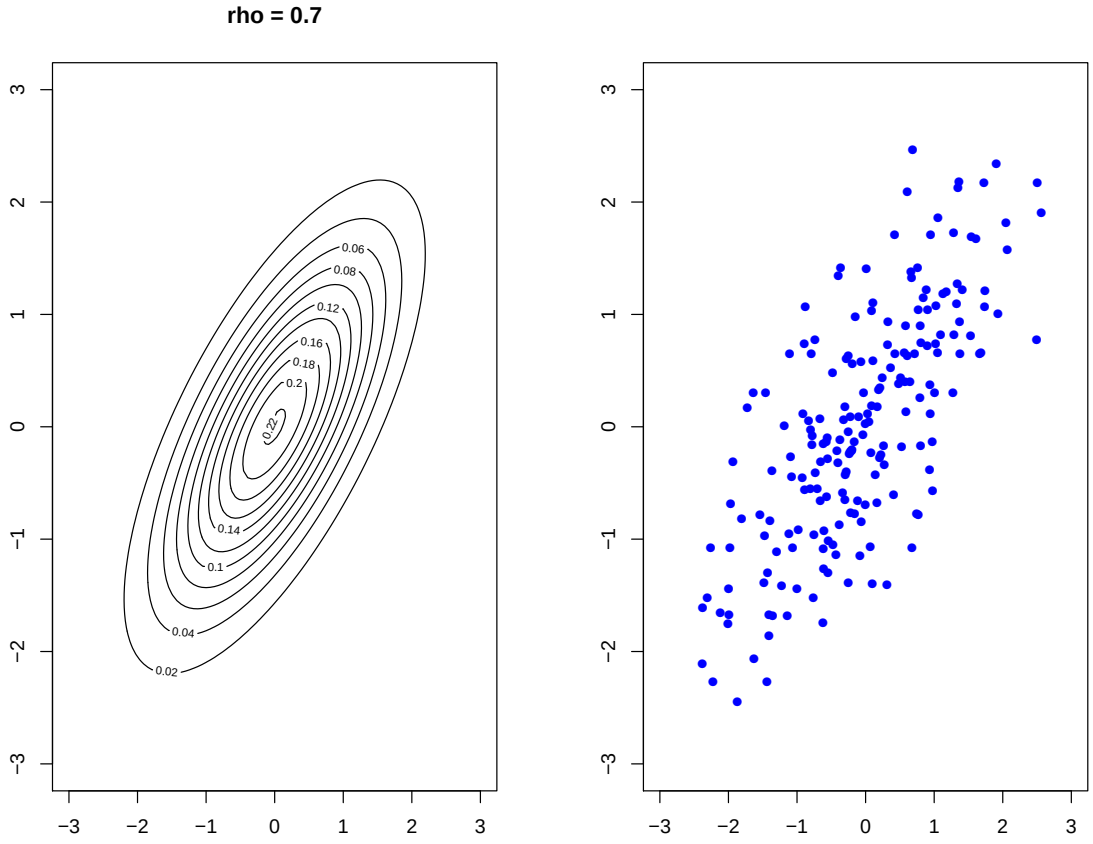
Multivariate Normal Density

$$X \sim N_2(\mu, \mathbf{\Sigma}) \text{ with } \mu = (0,0)^T \text{ and } \mathbf{\Sigma} = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$$



Multivariate Normal Density

Density (left) and Simulated Data from an MVN (right)



A quick recap of vector calculus (I)

Let a and x be p-column vectors.

If $y = y(x)$ is a scalar function of p arguments, i.e. of a p-column vector x, then the vector of partial derivatives $\left(\frac{\partial y}{\partial x_1}, \frac{\partial y}{\partial x_2}, \ldots, \frac{\partial y}{\partial x_p}\right)^T$ is called a gradient of y wrt x and denoted $\nabla_x y$

For example let $y = a^T x = \sum_{j=1}^{p} a_j x_j$ be a scalar function, then

$$\nabla_x y = \left(\frac{\partial y}{\partial x_1}, \frac{\partial y}{\partial x_2}, \ldots, \frac{\partial y}{\partial x_p}\right)^T = (a_1, a_2, \ldots, a_p)^T = a$$

A quick recap of vector calculus (II)

x is a p-column vector and $\boldsymbol{B}$ is a $p \times p$ symmetric matrix with rows $\boldsymbol{B}_1^T, \boldsymbol{B}_2^T \ldots, \boldsymbol{B}_p^T$.

Let $z = x^T \boldsymbol{B} x = \sum_{j=1}^{p} \sum_{k=1}^{p} \boldsymbol{B}_{jk} x_j x_k$. Symmetry implies that $\boldsymbol{B}_{jk} = \boldsymbol{B}_{kj}$.

Then consider

$$\nabla_x z = \left(\frac{\partial z}{\partial x_1}, \frac{\partial z}{\partial x_2}, \ldots, \frac{\partial z}{\partial x_p} \right)^T$$

Since $\frac{\partial z}{\partial x_j} = 2 \sum_{k=1}^{p} \boldsymbol{B}_{jk} x_k = 2 \boldsymbol{B}_j^T x$ we have that

$$\nabla_x z = 2 \boldsymbol{B} x$$

The matrix equivalent

Similarly we can consider scalar functions of *matrix arguments*, say $y = y(\boldsymbol{X})$ where $\boldsymbol{X}$ is an $n \times n$ matrix. Then there is also $n \times n$ partial derivatives and the matrix of partial derivatives of a scalar y function of with respect to the matrix $\boldsymbol{X}$ forms the gradient $\nabla_{\boldsymbol{X}} y$

$$\nabla_{\boldsymbol{X}} y = \begin{bmatrix} \frac{\partial y}{\partial x_{11}} & \frac{\partial y}{\partial x_{12}} & \cdots & \frac{\partial y}{\partial x_{1n}} \\ \frac{\partial y}{\partial x_{21}} & \frac{\partial y}{\partial x_{22}} & \cdots & \frac{\partial y}{\partial x_{2n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y}{\partial x_{n1}} & \frac{\partial y}{\partial x_{n2}} & \cdots & \frac{\partial y}{\partial x_{nn}} \end{bmatrix}.$$

Estimating parameters for the multivariate normal distribution

Let $x_1, \ldots, x_n$ be a random sample from a $N_p(\mu, \mathbf{\Sigma})$ distribution so there are n random vectors which are all i.i.d., but within each vector there are dependencies between dimensions

(We will do this only partially here. The full derivation is in exercises.)

The maximum likelihood estimates of μ and $\mathbf{\Sigma}$ are

$$\widehat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i \text{ and } \widehat{\mathbf{\Sigma}} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \widehat{\mu})(x_i - \widehat{\mu})^T.$$

Note that $(x_i - \widehat{\mu})$ is a $p \times 1$ column vector and so $(x_i - \widehat{\mu})^T$ is a $1 \times p$ row vector and so $\widehat{\mathbf{\Sigma}}$ is an $p \times p$ matrix.

The log-likelihood function is

$$\begin{aligned}
\ell(\mu, \mathbf{\Sigma}) &= \log \prod_{i=1}^{n} \frac{1}{(2\pi)^{p/2}|\mathbf{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(x_i - \mu)^T \mathbf{\Sigma}^{-1}(x_i - \mu)\right) \\
&= \text{constant} - \frac{n}{2}\log|\mathbf{\Sigma}| - \frac{1}{2}\sum_{i=1}^{n}(x_i - \mu)^T \mathbf{\Sigma}^{-1}(x_i - \mu) \\
&= -\frac{n}{2}\log|\mathbf{\Sigma}| - \frac{1}{2}\sum_{i=1}^{n}(x_i^T \mathbf{\Sigma}^{-1} x_i - 2x_i^T \mathbf{\Sigma}^{-1}\mu + \mu^T \mathbf{\Sigma}^{-1}\mu)
\end{aligned}$$

Then the vector of partial derivatives wrt μ is given by

$$\nabla_\mu \ell = -\frac{1}{2}\sum_{i=1}^{n}(-2\mathbf{\Sigma}^{-1}x_i + 2\mathbf{\Sigma}^{-1}\mu) \quad \text{by recap result}$$

Setting to zero leads to

$$\widehat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

The estimator of $\boldsymbol{\Sigma}$ is biased (remember : not all MLEs are unbiased). An unbiased estimator for $\boldsymbol{\Sigma}$ is

$$S = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \widehat{\mu})(x_i - \widehat{\mu})^T$$

We refer to S as the **sample covariance matrix**. If X is 'mean centered' (by subtraction of $\widehat{\mu}^T$ from each row), then

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

Sometimes it is useful to work with the **sample correlation matrix**, denoted R,

$$R_{jk} = \frac{S_{jk}}{\sqrt{S_{jj}S_{kk}}}$$

Crabs: Sample covariance matrix

Each specimen has measurements on:

- the width of the frontal lobe (FL),
- the rear width (RW),
- the length along the carapace midline (CL),
- the maximum width (CW) of the carapace,
- the body depth (BD) in mm.

On the Crabs data the sample covariance matrix is

$$S = \begin{array}{c|ccccc} & FL & RW & CL & CW & BD \\ \hline FL & 12.21 & 8.15 & 24.35 & 26.55 & 11.82 \\ RW & 8.15 & 6.62 & 16.35 & 18.23 & 7.83 \\ CL & 24.35 & 16.35 & 50.67 & 55.76 & 23.97 \\ CW & 26.55 & 18.23 & 55.76 & 61.96 & 26.09 \\ BD & 11.82 & 7.83 & 23.97 & 26.09 & 11.72 \end{array}.$$

Crabs: Sample correlation matrix

On the Crabs data the sample correlation matrix is

$$\boldsymbol{R} = \begin{array}{c|ccccc} & FL & RW & CL & CW & BD \\ \hline FL & 1.00 & 0.91 & 0.98 & 0.96 & 0.99 \\ RW & 0.91 & 1.00 & 0.89 & 0.90 & 0.89 \\ CL & 0.98 & 0.89 & 1.00 & 1.00 & 0.98 \\ CW & 0.96 & 0.90 & 1.00 & 1.00 & 0.97 \\ BD & 0.99 & 0.89 & 0.98 & 0.97 & 1.00 \end{array}.$$

Principal Components Analysis (PCA)

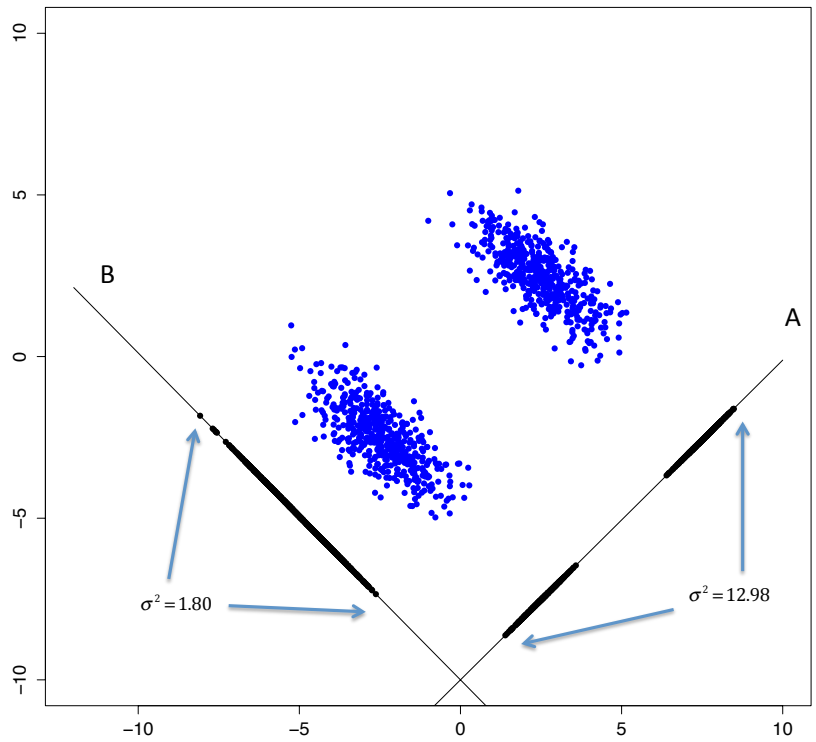
PCA aims to find a useful low-dimensional representation of the data, i.e. to construct a mapping $x_i \mapsto z_i$, $x_i \in \mathbb{R}^p$ and $z_i \in \mathbb{R}^q$ with $p > q$ from original variables to transformed variables
Question 'how do we define a representative variable or dimension?'
Key ideas

1. Each of the transformed variables found by PCA is a linear combination of the variables.
2. PCA seeks a small number of transformed variables that are as *interesting* as possible, where interesting is measured by how much the observations vary once transformed. These dimensions are refered to as **principal components**, or PCs for short.

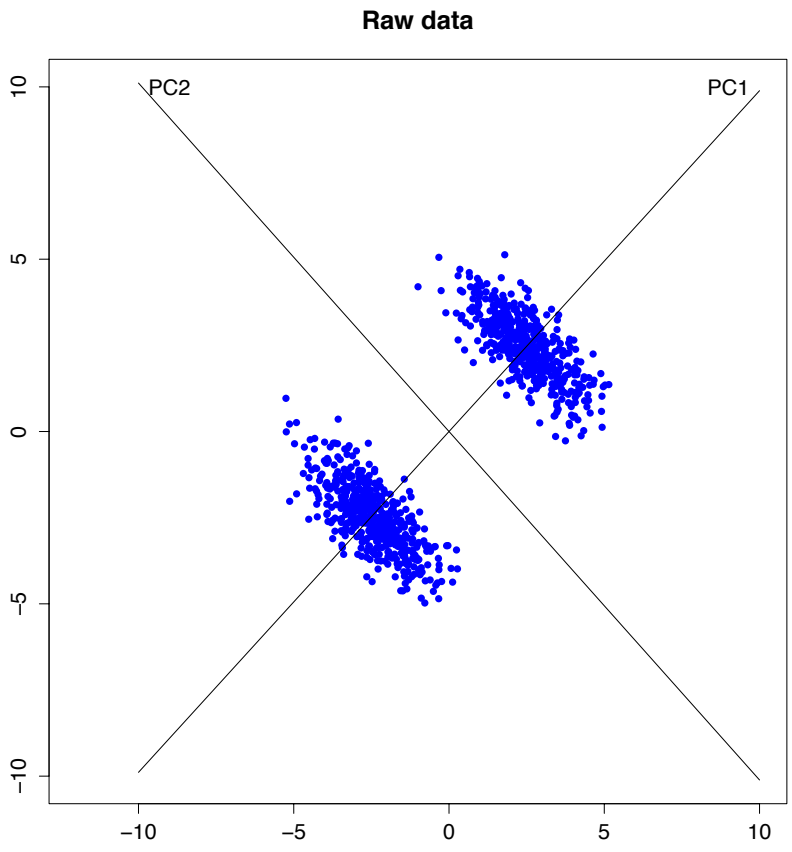
Projections that maximize variance can (but not always) find useful structure in datasets.
PCA finds a set of orthogonal projections of the dataset.

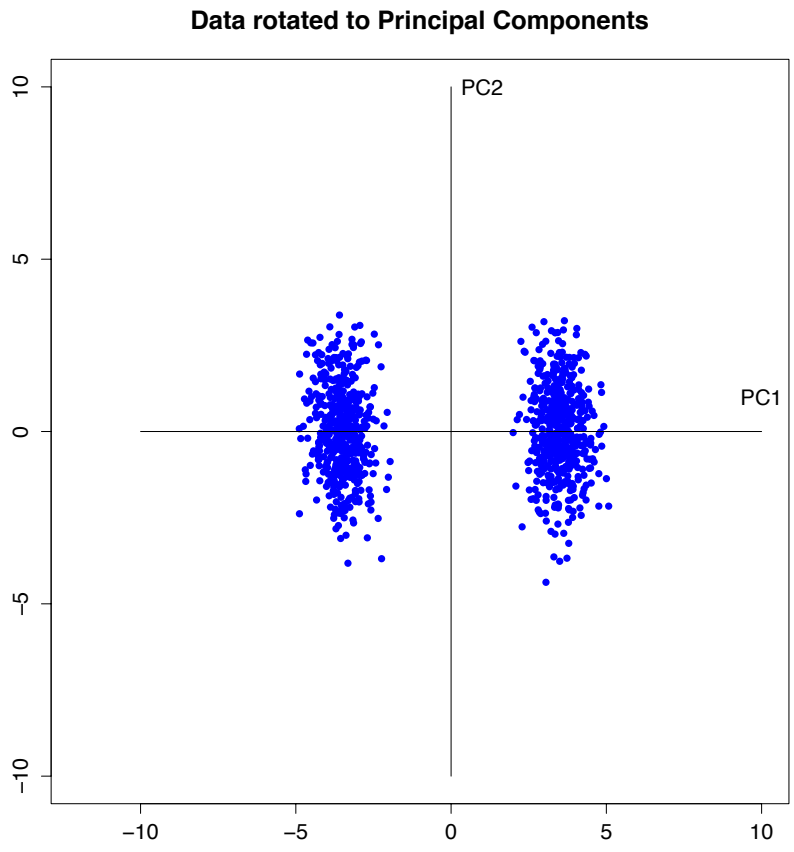
PCA

Projections that maximize variance can find useful structure in datasets. Projecting onto A separates clusters and has higher variance that projecting onto B.



PCA


Raw data
PC2
PC1
10
5
0
−5
−10
−10
−5
0
5
10


Data rotated to Principal Components
PC2
PC1
10
5
0
−5
−10
−10
−5
0
5
10

Finding the first principal component

The PCA components are found one at a time. The first principal component is a linear combination of the set of p variables, denoted Z_1, such that

$$Z_1 = \alpha_{11} X_1 + \alpha_{12} X_2 + \ldots + \alpha_{1p} X_p$$

We can write

$$Z_1 = \alpha_1^T X$$

where $\alpha_1 = (\alpha_{11}, \alpha_{12}, \ldots, \alpha_{1p})^T$ and $X = (X_1, \ldots, X_p)^T$ are both column vectors. Then it can be shown (Exercise Sheet) that

$$\text{var}(Z_1) = \alpha_1^T \mathbf{\Sigma} \alpha_1$$

where $\mathbf{\Sigma}$ is the $p \times p$ covariance matrix of the random variable X.

Two problems

1. Since we only have a sample of data from each of the p-variables we do not know the true covariance $\mathbf{\Sigma}$, but we can use the sample covariance matrix $\boldsymbol{S}$ instead.
2. This variance is unbounded as the α_j's increase, so add a constraint on α such that

$$\sum_{j=1}^{p} \alpha_{1j}^2 = \alpha_1^T \alpha_1 = 1$$

This results in the constrained maximization problem

$$\max_{\alpha_1} \alpha_1^T \boldsymbol{S} \alpha_1 \quad \text{subject to} \quad \alpha_1^T \alpha_1 = 1$$

We can solve this using Lagrange Multipliers (see Prelims Intro Calculus course for a reminder on this technique).
Let

$$\mathcal{L}(\alpha_1, \lambda_1) = \alpha_1^T \boldsymbol{S} \alpha_1 - \lambda_1(\alpha_1^T \alpha_1 - 1)$$

We then need the vector of partial derivatives of $\mathcal{L}$ with respect to the vector α_1. This is the gradient vector $\nabla_{\alpha_1}\mathcal{L}$ seen in Intro Calculus.

Re-writing we have

$$\mathcal{L}(\alpha_1, \lambda_1) = \alpha_1^T \boldsymbol{S} \alpha_1 - \lambda_1 \alpha_1^T \boldsymbol{I}_p \alpha_1 + \lambda_1$$

where $\boldsymbol{I}_p$ denotes the $p \times p$ identity matrix.
Using these results we have that

$$\nabla_{\alpha_1}\mathcal{L}(\alpha_1, \lambda_1) = 2\boldsymbol{S}\alpha_1 - 2\lambda_1 \boldsymbol{I}_p \alpha_1 = 2\boldsymbol{S}\alpha_1 - 2\lambda_1\alpha_1 \quad (\text{p} - \text{dimensional})$$

$$\nabla_{\lambda_1}\mathcal{L}(\alpha_1, \lambda_1) = -\alpha_1^T \boldsymbol{I}_p \alpha_1 + 1 \quad (1 - \text{dimensional})$$

Setting this equal to 0 results in the eigenvalue equation

$$\boldsymbol{S}\alpha_1 = \lambda_1\alpha_1 \tag{1}$$

Recap of results from Prelims Linear Algebra

Let V be a vector space over $\mathbb{R}$.

Let $\boldsymbol{A} \in M_n(\mathbb{R})$ be a linear transformation $\boldsymbol{A} : V \to V$.

Consider the equation $\boldsymbol{A}v = \lambda v$
If $v \neq 0$, $\lambda \in \mathbb{R}$ is a solution then we call v an eigenvector and λ an eigenvalue.

If $\boldsymbol{A}$ is a real symmetric matrix $\boldsymbol{A} \in M_n(\mathbb{R})$ then all its eigenvalues are real and we can write

$$\boldsymbol{A} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T$$

where columns of $\boldsymbol{V}$ contain the eigenvectors of $\boldsymbol{A}$ and $\boldsymbol{D}$ is a diagonal matrix of corresponding eigenvalues.

It is normal to assume that $\boldsymbol{D}$ contains a decreasing set of eigenvalues. We refer to this as the **eigendecomposition** of $\boldsymbol{A}$.

So λ_1 and α_1 will be an eigenvalue and eigenvector of $\boldsymbol{S}$ respectively.

The eigendecomposition of $\boldsymbol{S}$ is

$$\boldsymbol{S} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T$$

Which eigenvector/eigenvalue pair to choose?

If we multiply the equation ($\boldsymbol{S}\alpha_1 = \lambda_1\alpha_1$) by α_1^T then we get

$$\alpha_1^T \boldsymbol{S} \alpha_1 = \lambda_1 \tag{2}$$

Thus, λ_1 is the sample variance of the first principal component which we are trying to maximize, so we choose λ_1 to be the *largest* eigenvalue of $\boldsymbol{S}$, and α_1 is the corresponding eigenvector.

Finding the second principal component (I)

The first PC

$$Z_1 = \alpha_1^T X$$

is obtained by setting α_1 to the top eigenvector of the sample covariance matrix S, i.e. it satisfies $S\alpha_1 = \lambda_1 \alpha_1$ where λ_1 is the sample variance of Z_1.

The second principal component will also be a linear combination of the set of p variables, denoted Z_2, such that

$$Z_2 = \alpha_2^T X = \alpha_{21} X_1 + \alpha_{22} X_2 + \ldots + \alpha_{2p} X_p$$

where $\alpha_2 = (\alpha_{21}, \alpha_{22}, \ldots, \alpha_{2p})^T$.
As before we want to maximize $\alpha_2^\top \boldsymbol{S} \alpha_2$ and need the constraint

$$\alpha_2^T \alpha_2 = 1$$

We need another constraint to find a distinct eigenvector.

Finding the second principal component (II)

We wish the linear combinations of p variables to be orthogonal projections, so we add in the further constraint that

$$\alpha_1^T \alpha_2 = 0$$

and this leads to another Lagrange multipliers problem where we seek to maximize

$$\mathcal{L}(\alpha_2, \lambda_2, m) = \alpha_2^T \mathbf{S} \alpha_2 - \lambda_2(\alpha_2^T \alpha_2 - 1) - m\alpha_1^T \alpha_2$$

Taking partial derivatives with respect to α_2 leads to

$$\nabla_{\alpha_2} \mathcal{L}(\alpha_2, \lambda_2, m) = 2\mathbf{S}\alpha_2 - 2\lambda_2 \alpha_2 - m\alpha_1$$

Finding the second principal component (III)

Setting equal to 0 gives

$$\boldsymbol{S}\alpha_2 - \lambda_2\alpha_2 = \frac{1}{2}m\alpha_1 \tag{3}$$

Pre-multiplying by α_1^T, remembering that $\alpha_1^T\alpha_2 = 0$ and $\alpha_1^T\alpha_1 = 1$, gives

$$\alpha_1^T\boldsymbol{S}\alpha_2 = \frac{1}{2}m$$

Since $\alpha_1^T\boldsymbol{S}\alpha_2$ is a scalar and $\boldsymbol{S}$ is symmetric and α_1 is its eigenvector, we get

$$\frac{1}{2}m = \alpha_1^T\boldsymbol{S}\alpha_2 = \alpha_2^T\boldsymbol{S}\alpha_1 = \lambda_1\alpha_2^T\alpha_1 = 0, \tag{4}$$

which implies that $m = 0$ and equation (3) reduces to the eigenvalue equation

$$\boldsymbol{S}\alpha_2 = \lambda_2\alpha_2 \tag{5}$$

So λ_2 is also an eigenvalue of $\boldsymbol{S}$ with associated eigenvector α_2.

Finding the second principal component (IV)

Multiplying by α_2^T gives

$$\alpha_2^T \mathbf{S} \alpha_2 = \lambda_2 \tag{6}$$

So λ_2 is equal to $\alpha_2^T \mathbf{S} \alpha_2$ i.e. sample variance of the second principal component.

We must also ensure that $\alpha_1^T \alpha_2 = 0$, so this is achieved by setting λ_2 to be the second largest eigenvalue, with α_2 being its corresponding eigenvector.

Finding subsequent principal components (I)

The above process can be continued for the other principal components. This results in a sequence of principal components ordered by their variance. In other words, if we consider the eigenvalue decomposition of $\boldsymbol{S}$ (see Linear Algebra recap)

$$\boldsymbol{S} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T$$

where $\boldsymbol{D}$ is a diagonal matrix of eigenvalues ordered in decreasing value, and $\boldsymbol{V}$ is a $p \times p$ matrix of the corresponding eigenvectors as orthonormal columns $(v_1, \ldots, v_p)$, then we have that

$$\alpha_j = v_j \text{ and } \lambda_j = \boldsymbol{D}_{jj} \text{ for } j = 1, \ldots, p$$

Finding subsequent principal components (II)

Once we have eigendecomposition, we can compute the value of the ℓ-th principal component for i-th data vectors, i.e.

$$z_{i\ell} = \alpha_\ell^T x_i = \alpha_{\ell 1} x_{i1} + \cdots + \alpha_{\ell p} x_{ip}$$

For $\ell = 1, \ldots, q$, the new transformed dataset gives matrix $Z \in \mathbb{R}^{n \times q}$, with n: data vectors, q: reduced dimension.

Centering

Since PCA is interested in modelling variances and covariances, it is customary to work with mean-centred data. I.e. we first subtract sample mean from the data vectors in order to simplify computations. We take sample mean $\widehat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and subtract it from the data vectors, i.e. we use

$$\widetilde{x}_i = x_i - \widehat{\mu}.$$

Thus, we will assume this was already done, such that $\frac{1}{n}\sum_{i=1}^{n} x_i = 0$.

In that case we can write sample covariance as

$$\boldsymbol{S} = \frac{1}{n-1}\boldsymbol{X}^T\boldsymbol{X}$$

Note then for the ℓ-th PC we have

$$\frac{1}{n}\sum_{i=1}^{n} z_{i\ell} = \alpha_\ell^T \left(\frac{1}{n}\sum_{i=1}^{n} x_i \right) = 0,$$

so transformed data is also centred, and

$$\begin{aligned}
\alpha_\ell^T \boldsymbol{S} \alpha_\ell &= \frac{1}{n-1} \alpha_\ell^T \boldsymbol{X}^T \boldsymbol{X} \alpha_\ell = \frac{1}{n-1} (\boldsymbol{X}\alpha_\ell)^T \boldsymbol{X}\alpha_\ell \\
&= \frac{1}{n-1} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} x_{ij}\alpha_{\ell j} \right)^2 = \frac{1}{n-1} \sum_{i=1}^{n} (z_{i\ell})^2 \\
&= \lambda_\ell,
\end{aligned}$$

which is sample variance of $\{z_{i\ell}\}_{i=1}^n$ (ℓ-th dimension in the transformed data).

Plotting the principal components

$\boldsymbol{X}$ is the $n \times p$ data matrix
$\boldsymbol{S}$ is the $p \times p$ sample covariance matrix with $\boldsymbol{S} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T$
Define $\boldsymbol{Z}$ to be an $n \times p$ matrix containing the transformed data, such that $\boldsymbol{Z}_{ij}$ is the value of the jth principal component for the ith observation.

Then we have

$$\boldsymbol{Z}_{ij} = x_i^T \alpha_j = x_i^T v_j$$

i.e. the vector product of the ith row of $\boldsymbol{X}$ and the jth column of $\boldsymbol{V}$.
In matrix notation we can write this as

$$\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{V}$$

The matrix $\boldsymbol{V}$ is known as the **loadings matrix**.

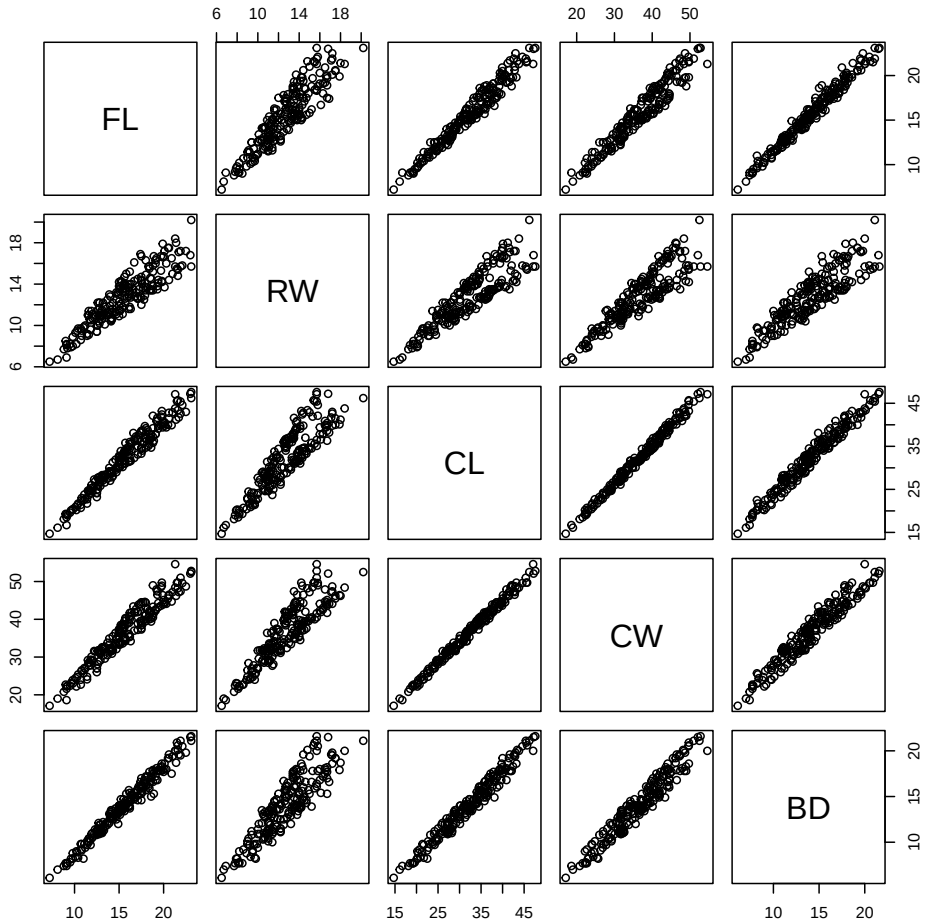
Then can make the pairwise plot the columns of $\boldsymbol{Z}$ against each other to visualise the data as represented by those pairs of principal components.

The matrix $\boldsymbol{Z}$ is known as the **scores matrix**.

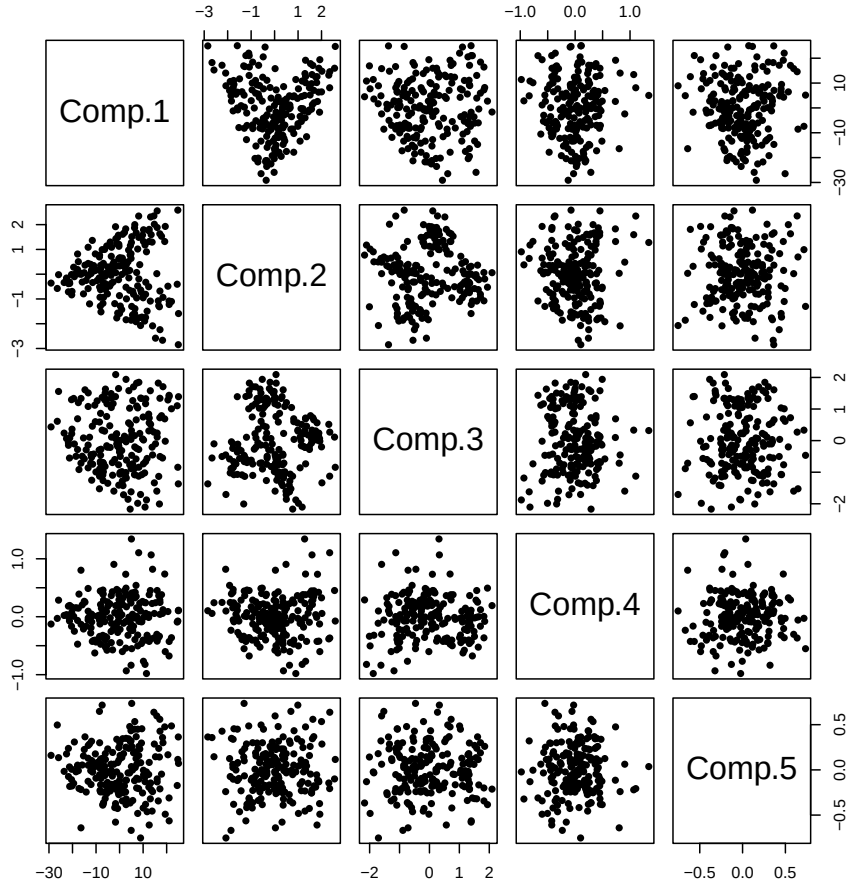
The columns of this matrix contain the projections onto individual principal components.

Crabs: Pairs plots

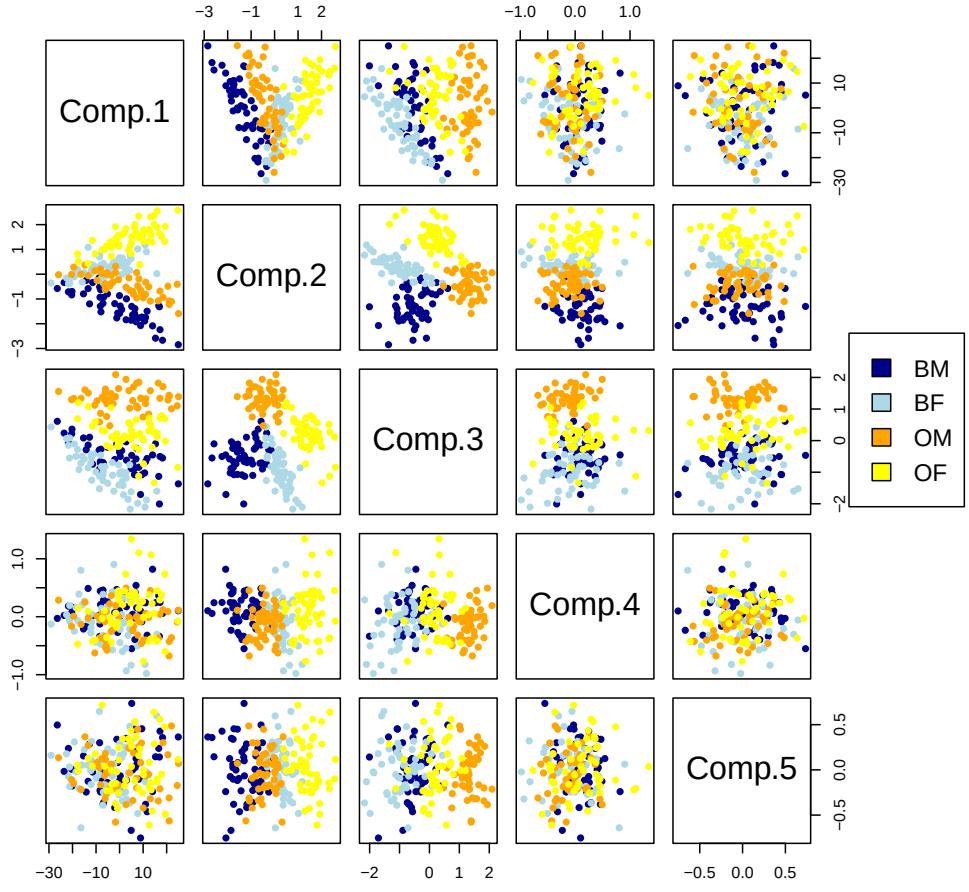
Plotting pairs of variables together in a scatter plot can be helpful to see how variables co-vary.



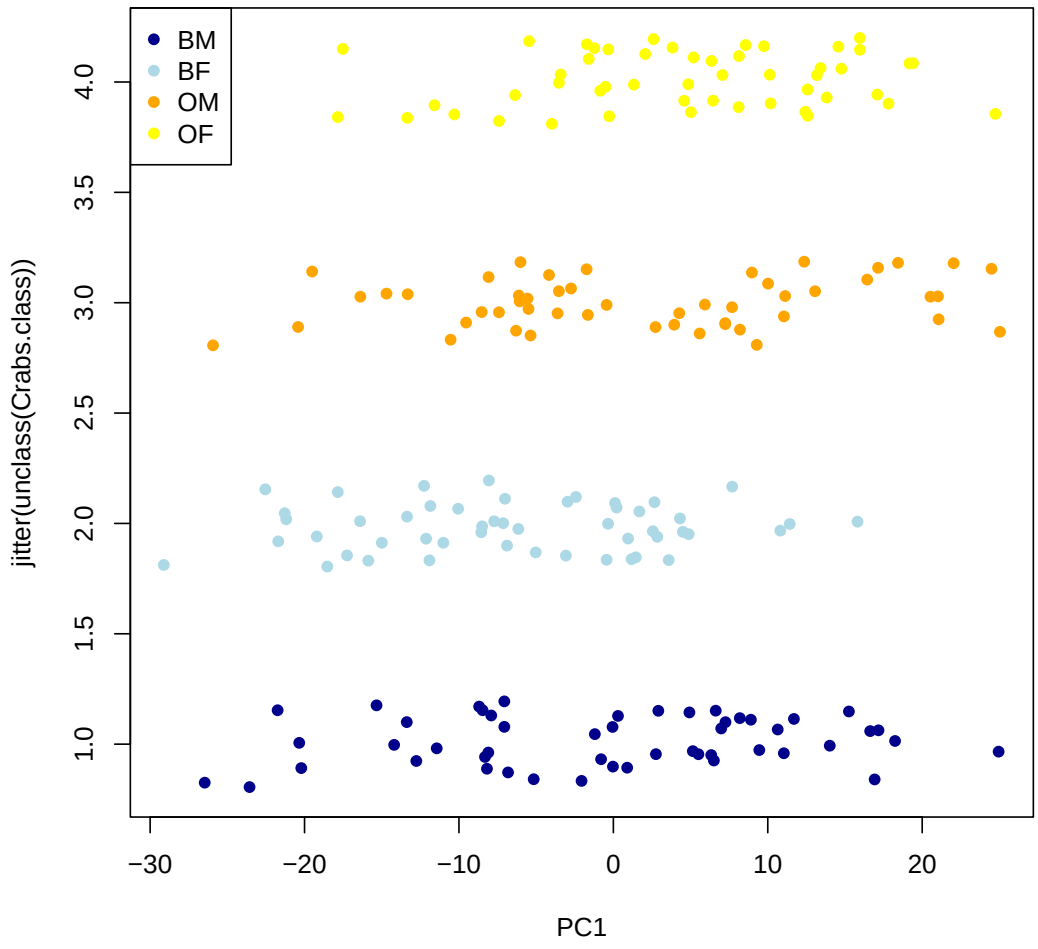
Pairs plot of PCA of Crabs dataset



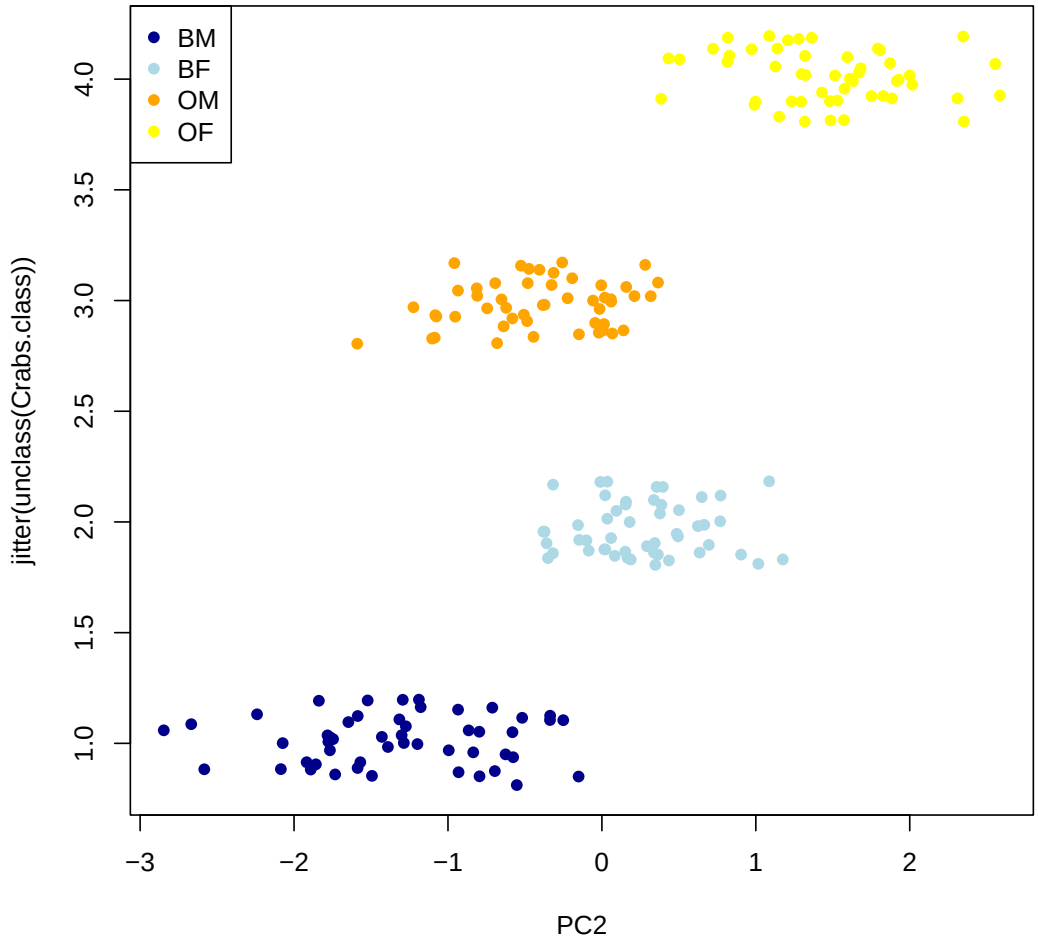
Pairs plot of PCA of Crabs dataset



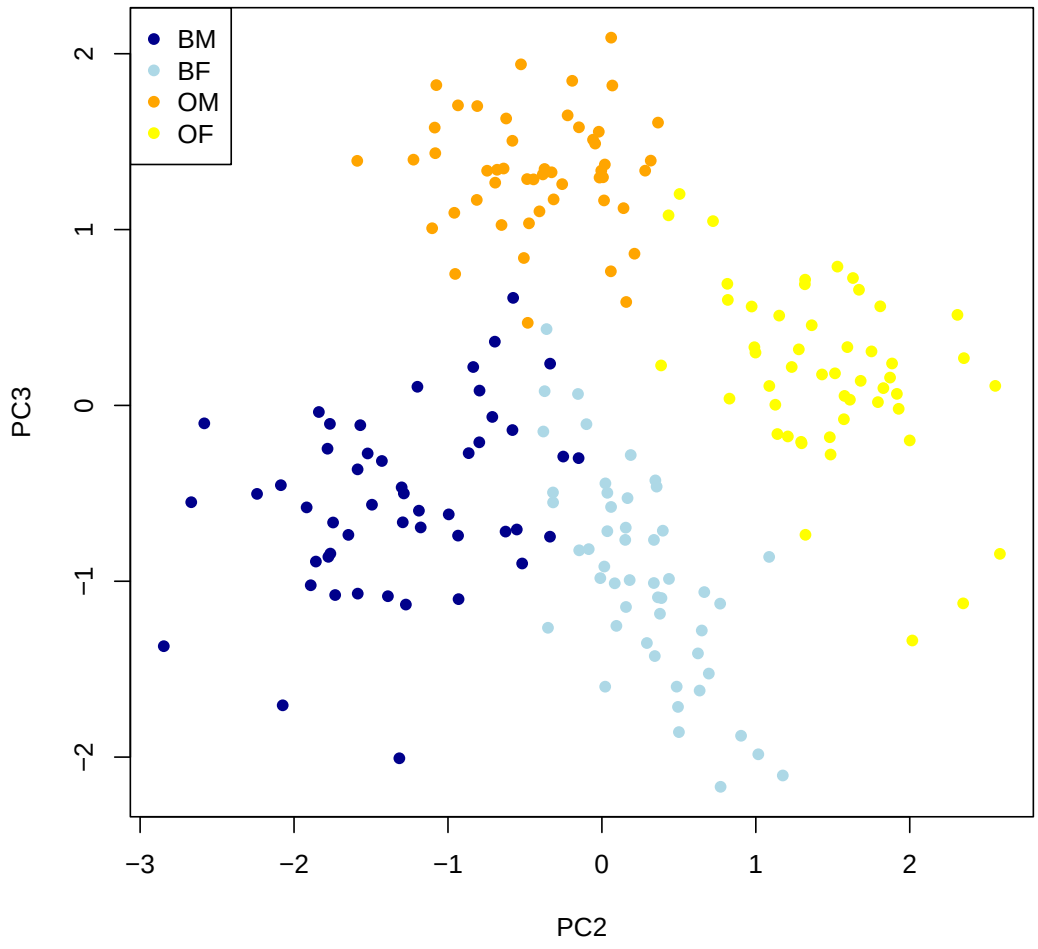
PC1 for the Crabs dataset



PC2 for the Crabs dataset



PC2 vs PC3 for the Crabs dataset



Loadings for the Crabs dataset

$$\boldsymbol{V} = \begin{array}{c|ccccc} & PC1 & PC2 & PC3 & PC4 & PC5 \\ \hline FL & \mathbf{0.28} & \mathbf{0.32} & \mathbf{0.50} & 0.73 & 0.12 \\ RW & \mathbf{0.19} & \mathbf{0.86} & \mathbf{-0.41} & -0.14 & -0.14 \\ CL & \mathbf{0.59} & \mathbf{-0.19} & \mathbf{0.17} & -0.14 & -0.74 \\ CW & \mathbf{0.66} & \mathbf{-0.28} & \mathbf{-0.49} & 0.12 & 0.47 \\ BD & \mathbf{0.28} & \mathbf{0.15} & \mathbf{0.54} & -0.63 & 0.43 \end{array}$$

So for example, this means that the first, second and third PCs are

$$Z_1 = \mathbf{0.28}FL + \mathbf{0.19}RW + \mathbf{0.59}CL + \mathbf{0.66}CW + \mathbf{0.28}BD$$

$$Z_2 = \mathbf{0.32}FL + \mathbf{0.86}RW\mathbf{-0.19}CL\mathbf{-0.28}CW + \mathbf{0.15}BD$$

$$Z_3 = \mathbf{0.50}FL\mathbf{-0.41}RW + \mathbf{0.17}CL\mathbf{-0.49}CW + \mathbf{0.54}BD$$

Biplots

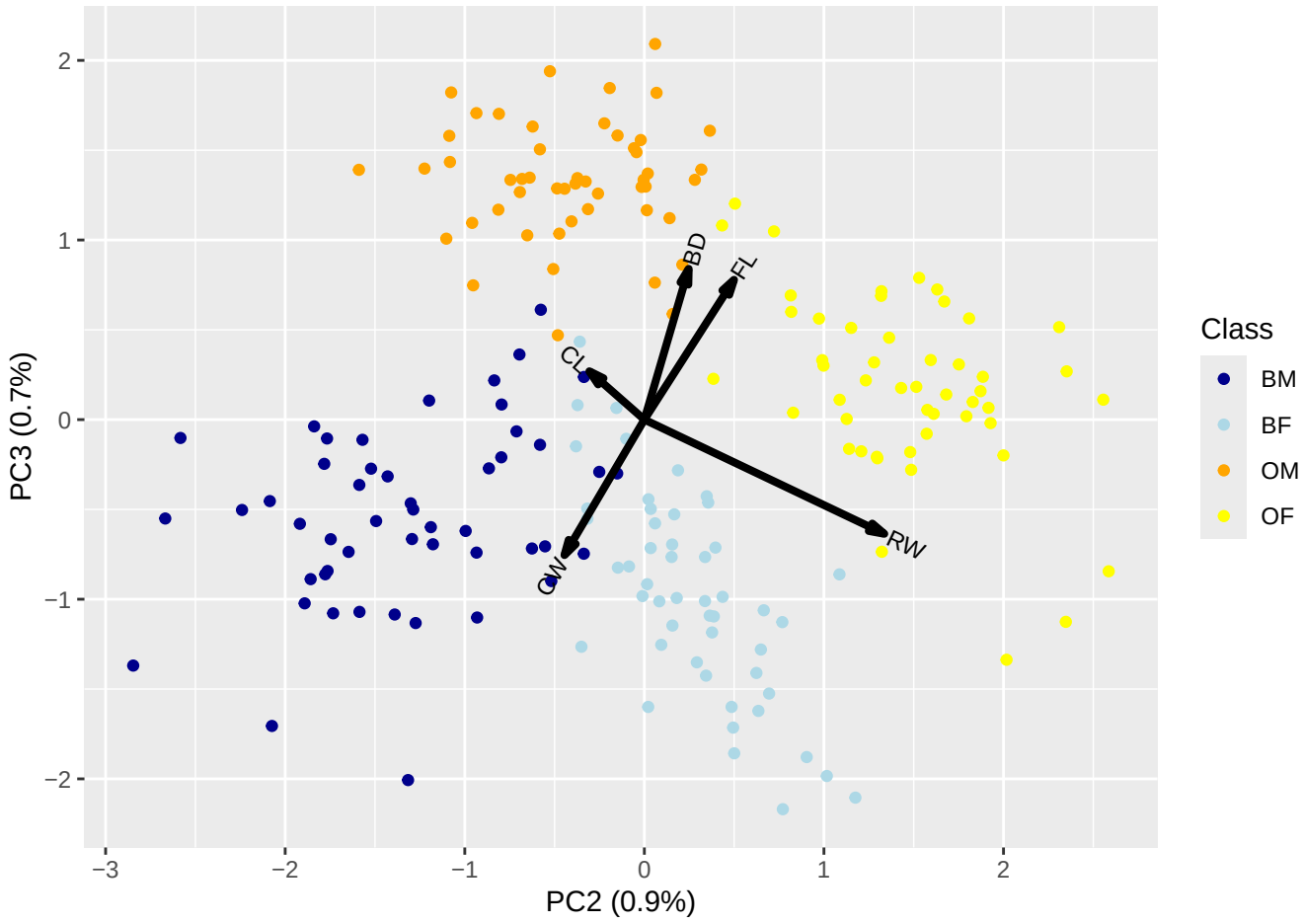
The columns of the loadings matrix $\boldsymbol{V}$ contains the coefficients appearing in expressions for each of the PCs and tell us how much original variables contribute to those PCs. It can be interesting to look at these loadings to see which variables contribute to each PC.

It can be useful to represent the loadings information on the plots of the PCs scores.

FL has weight 0.32 in PC2 and 0.50 in PC3.
We represent that on the PC plot using a vector (0.32, 0.50).
Similarly, we can represent the contribution of variable RW on the PC plot using a vector (0.86, -0.41) etc.

Often we need to scale the vectors by a constant factor to appear on the plot in a visually appealing way.

BiPlot of PCs 2 and 3 for the Crabs dataset.



Variance decomposition and eigenspectrum (I)

We started with 5 variables and ended up considering three. (While the 1st PC explained most of the variance, it was irrelevant for the problem of splitting crabs into groups.) How do you decide how many PCs to use?

The total amount of variance in the original data matrix $\boldsymbol{X}$ is the sum of the diagonal entries in $\boldsymbol{S}$. In other words,

$$\text{Total variance } = \sum_{j=1}^{p} \boldsymbol{S}_{jj} = tr(\boldsymbol{S})$$

but since $tr(AB) = tr(BA)$ where A and B are two $p \times p$ matrices we have that

$$\text{Total variance } = tr(\boldsymbol{S}) = tr(\boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T) = tr(\boldsymbol{D}\boldsymbol{V}^T\boldsymbol{V}) = tr(\boldsymbol{D})$$

since $\boldsymbol{V}$ is an orthonormal matrix of eigenvectors, so that $\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}$.

Variance decomposition and eigenspectrum (II)

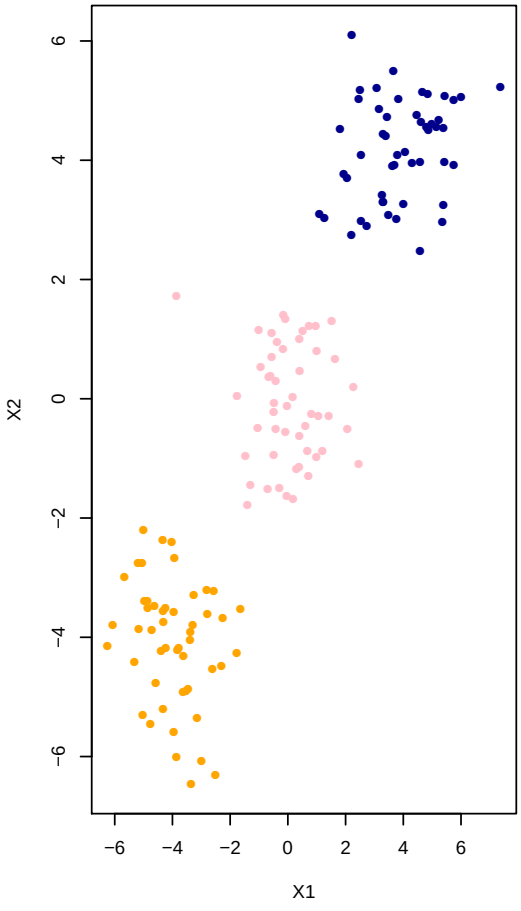
Thus, the total variance remains the same on transformed data (assuming we use all p PCs), it just is repackaged.

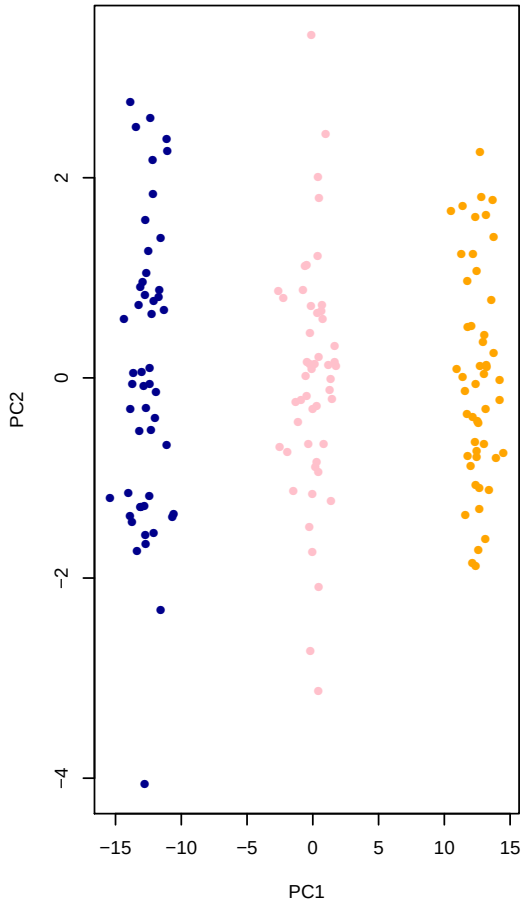
Plot the decreasing sequence of eigenvalues to visualise the structure in the dataset. Such plots are sometimes referred to as **eigenspectrum plots** or **variance scree plots**, and usually they are scaled so each bar is percentage of the total variance. That is we plot

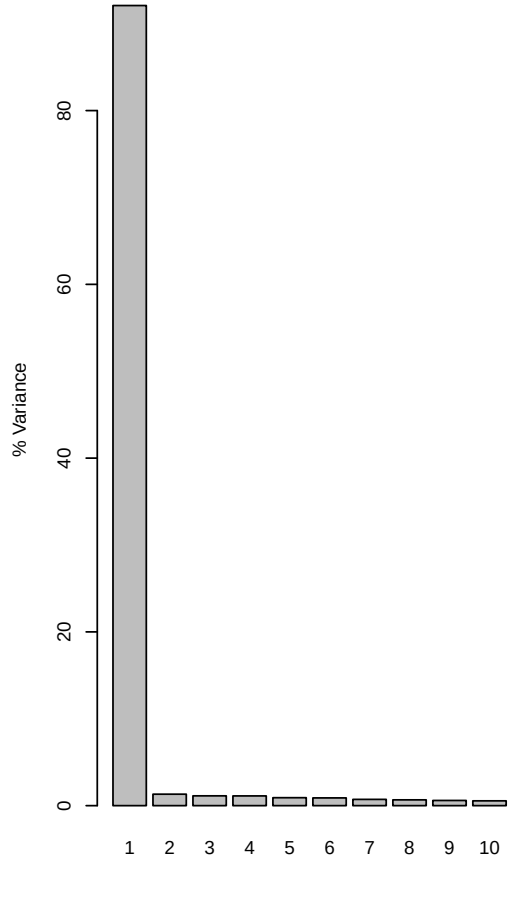
$$\frac{100\boldsymbol{D}_{jj}}{tr(\boldsymbol{D})} \quad \text{for } j \in 1, \ldots, p$$

The scree plot is sometimes used to decide on a number of PCs to carry forward for further analysis. For example, we might choose to take the PCs that account for the top $100(1-\alpha)\%$ of the variance.

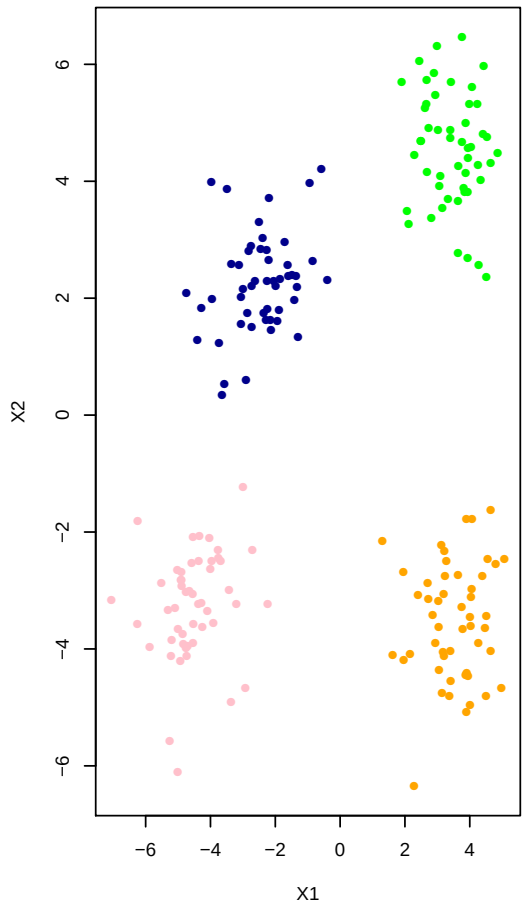
Scree plot example 1

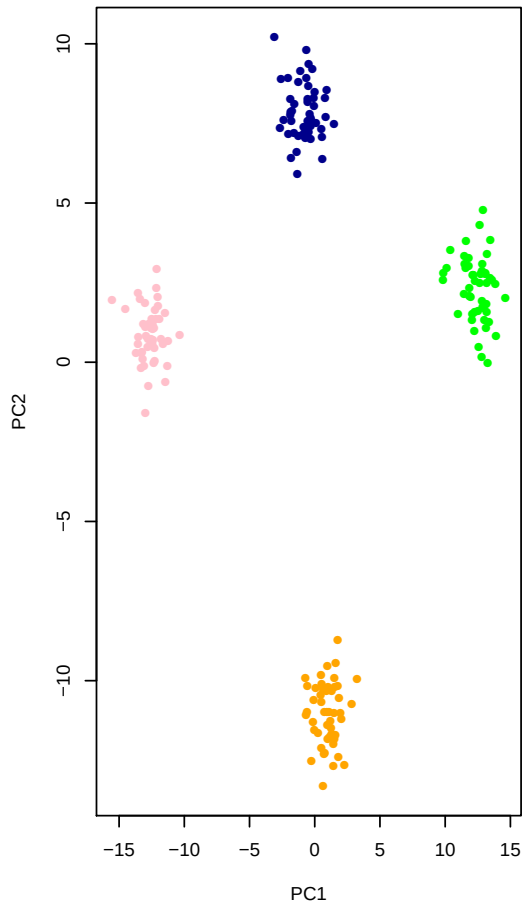


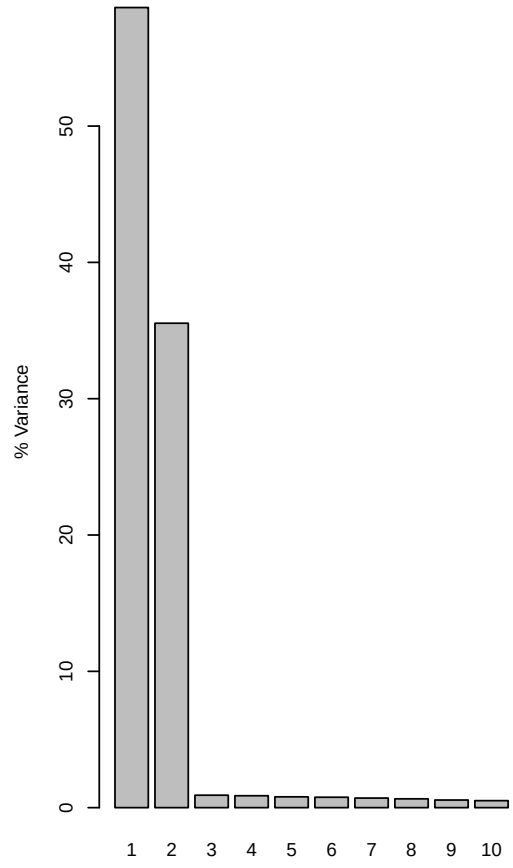




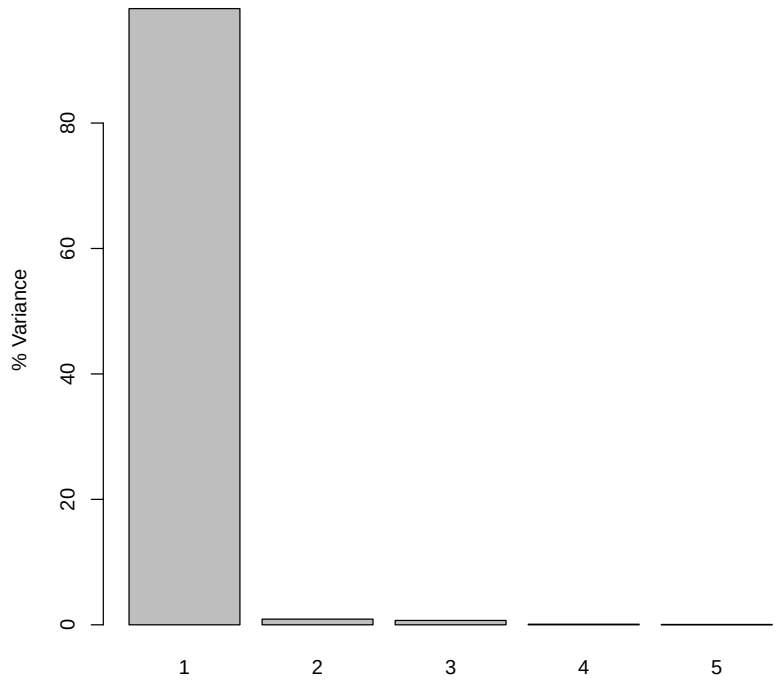
Scree plot example 2







Scree plot for Crabs dataset



Using the covariance matrix or the correlation matrix

Recall we had sample covariance matrix of a centred dataset:

$$\boldsymbol{S} = \frac{1}{n-1}\boldsymbol{X}^T\boldsymbol{X}$$

Each entry j,k corresponded to sample covariance between variables j and k

$$\boldsymbol{S}_{jk} = \frac{1}{n-1}\sum_{i=1}^{n} x_{ij}x_{ik}$$

which estimates $\text{cov}(X_j, X_k)$.

We then obtained PCA by eigendecomposition of $\boldsymbol{S}$.

To transform or not to transform - that is the question

A key practical problem when applying PCA is deciding exactly what data the method should be applied to. **Should we apply the method to the raw data, or should we first transform it in some way?** You should always ask yourself this question when starting out on a new data analysis.

The first principal component maximises the variance of a linear combination of variables. If some variables have very different levels of variability due to being measured on a very large scale then maximizing the projection with the largest variance may tend to dominate the first principal component. This might be unsatisfactory when looking for structure in the dataset. For this reason, it can sometimes make sense to standardize the variables before PCA, so that each variable has the same variance.

Using the sample correlation matrix instead

An alternative is to use the **sample correlation matrix** $\boldsymbol{R}$, rather than the sample covariance matrix $\boldsymbol{S}$.

Remember that

$$\text{cor}(X_j, X_k) = \frac{\text{cov}(\text{X}_\text{j}, \text{X}_\text{k})}{\sqrt{\text{var}(X_j)\text{var}(X_k)}} \in [-1, 1]$$

This means that the relationship between $\boldsymbol{R}$ and $\boldsymbol{S}$ is

$$\boldsymbol{R}_{jk} = \frac{\boldsymbol{S}_{jk}}{\sqrt{\boldsymbol{S}_{jj}\boldsymbol{S}_{kk}}}$$

Using the sample correlation matrix instead

If we let $\boldsymbol{W}$ be a diagonal matrix with entries $\boldsymbol{S}_{ii}$ for $i \in 1, \ldots, p$. In other words, $\boldsymbol{W}$ is the same as $\boldsymbol{S}$, but with all off-diagonal entries set to 0. Then in matrix notation we can write

$$\boldsymbol{R} = \boldsymbol{W}^{-1/2}\boldsymbol{S}\boldsymbol{W}^{-1/2}$$

It can be shown (see Exercises) that the PCA components derived from using $\boldsymbol{S}$ and $\boldsymbol{R}$ will be the same when the variances of the variables are all identical.

So standardize the variables before PCA, so that each variable has the same variance!

EU indicators dataset

Economic indicators for 27 EU countries (2011)

Country	CPI	UNE	INP	BOP	PRC	UN%
Belgium	116.03	4.77	125.59	908.60	6716.50	-1.60
Bulgaria	141.20	7.31	102.39	27.80	1094.70	3.50
CzechRep.	116.20	4.88	119.01	-277.90	2616.40	-0.60
Denmark	114.20	6.03	88.20	1156.40	7992.40	0.50
Germany	111.60	4.63	111.30	499.40	6774.60	-1.30
Estonia	135.08	9.71	111.50	153.40	2194.10	-7.70
Ireland	106.80	10.20	111.20	-166.50	6525.10	2.00
Greece	122.83	11.30	78.22	-764.10	5620.10	6.40
Spain	116.97	15.79	83.44	-280.80	4955.80	0.70
France	111.55	6.77	92.60	-337.10	6828.50	-0.90
Italy	115.00	5.05	87.80	-366.20	5996.60	-0.50
Cyprus	116.44	5.14	86.91	-1090.60	5310.30	-0.40
Latvia	144.47	12.11	110.39	42.30	1968.30	-3.60
Lithuania	135.08	11.47	114.50	-77.40	2130.60	-4.30
Luxembourg	118.19	3.14	85.51	2016.50	10051.60	-3.00
Hungary	134.66	6.77	115.10	156.20	1954.80	-0.10
Malta	117.65	4.15	101.65	359.40	3378.30	-0.60
Netherlands	111.17	3.23	103.80	1156.60	6046.00	-0.40
Austria	114.10	2.99	116.80	87.80	7045.50	-1.50
Poland	119.90	6.28	146.70	-74.80	2124.20	-1.00
Portugal	113.06	9.68	89.30	-613.40	4073.60	0.80
Romania	142.34	4.76	131.80	-128.70	1302.20	3.20
Slovenia	118.33	5.56	105.40	39.40	3528.30	1.80
Slovakia	117.17	9.19	156.30	16.00	2515.30	-2.10
Finland	114.60	5.92	101.00	-503.70	7198.80	-1.30
Sweden	112.71	6.10	100.50	1079.10	7476.70	-2.30
UnitedKingdom	120.90	6.11	90.36	-24.30	6843.90	-0.80
Variance	111.66	9.95	357.27	450057.15	5992520.48	7.12

CPI=consumer price index; UNE=unemployment; INP=industrial production; BOP=balance of payments; PRC=private final consumption expenditure; UN%=annual change in unemployment rate; source= EUROSTAT

PCA on covariance vs correlation matrix

When using the covariance matrix $\boldsymbol{S}$ the loadings of the 1st and 2nd PCs are

$$\begin{aligned} Z_1 &= -0.003CPI - 0.0004UNE - 0.0039INP + 0.121BOP + 0.993PRC - 0.00003UN\% \\ Z_2 &= 0.004CPI - 0.001UNE + 0.009INP + 0.992BOP - 0.121PRC - 0.0014UN\% \end{aligned}$$

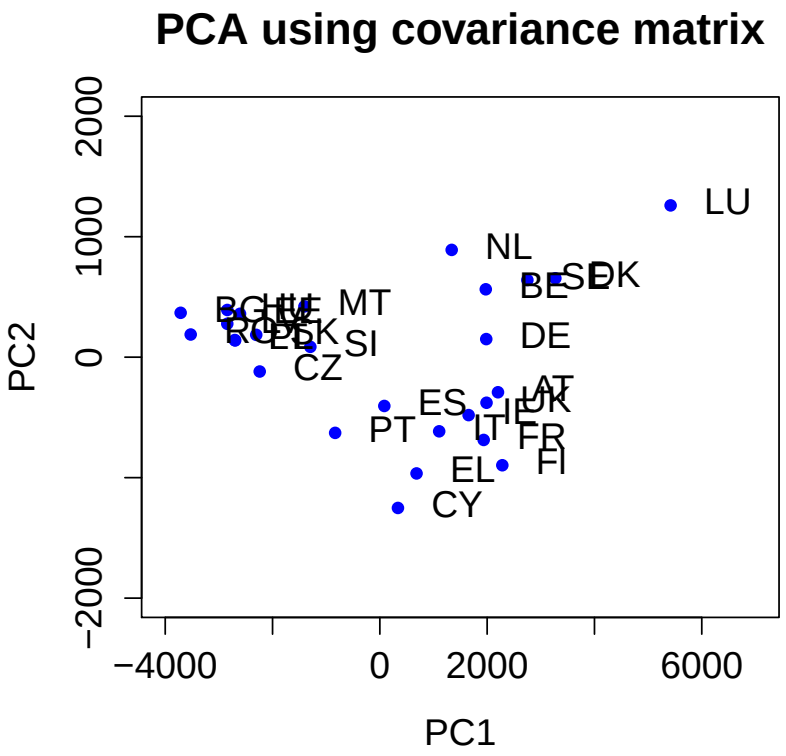
so it is the variables BOP and PRC that are dominating these PCs.

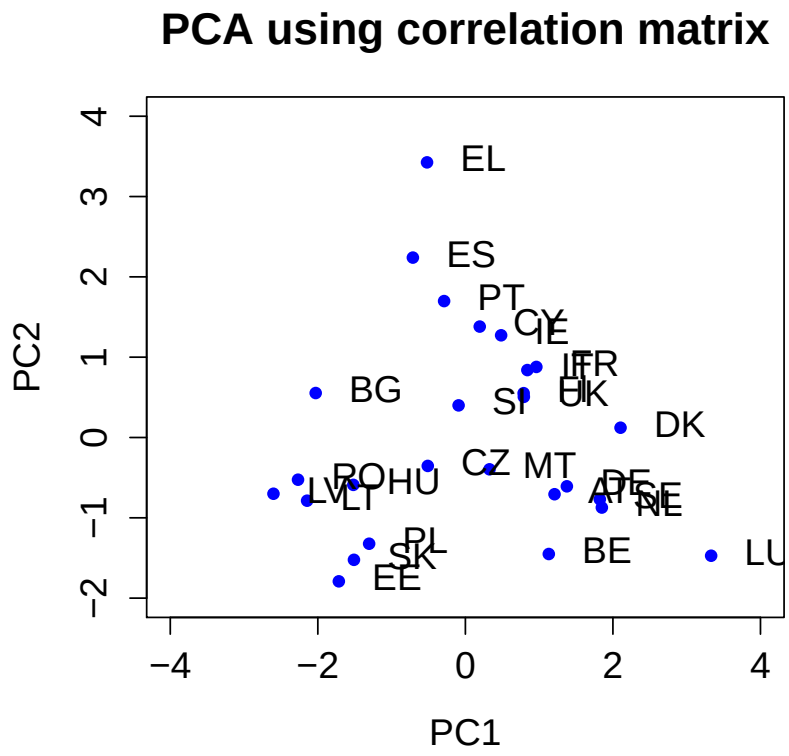
When using the correlation matrix $\boldsymbol{R}$ the loadings of the 1st and 2nd PCs are

$$\begin{aligned} Z_1 &= -0.51CPI - 0.37UNE - 0.29INP + 0.36BOP - 0.62PRC - 0.02UN\% \\ Z_2 &= -0.17CPI + 0.34UNE - 0.53INP - 0.49BOP + 0.12PRC + 0.56UN\% \end{aligned}$$

and the weightings for the variables are quite different.

PCA for EU indicators dataset





PCA via the Singular Value Decomposition (SVD)

Typical algorithms for finding the eigendecomposition of a $p \times p$ matrix $\boldsymbol{S}$ scale like $O(p^3)$. Also, when p is large (say much larger than n) this can be a very large matrix to store and work with on a computer. Also, since n points in a p-dimensional space defines a linear subspace whose dimension is at most $n - 1$, we would find that $p - n + 1$ eigenvalues are zero.

Faster way of obtaining the scores matrix $\boldsymbol{Z} = \boldsymbol{XV}$ is given by SVD

NOTE This theorem is given without proof and Prelims students are not expected to be able to prove it.

Theorem (Singular Value Decomposition) - I

If $\boldsymbol{X}$ is a $n \times p$ matrix of real numbers then there exists a factorization, called the Singular Value Decomposition (SVD) of $\boldsymbol{X}$, with the following form

$$\boldsymbol{X} = \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{Q}^T$$

where

- $\boldsymbol{P}$ is a $n \times n$ matrix such that $\boldsymbol{P}\boldsymbol{P}^T = \boldsymbol{P}^T\boldsymbol{P} = \boldsymbol{I}_n$
- $\boldsymbol{Q}$ is a $p \times p$ matrix such that $\boldsymbol{Q}\boldsymbol{Q}^T = \boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}_p$
- $\boldsymbol{\Lambda}$ is $n \times p$ with zeros off-diagonal and non-negative real numbers $\boldsymbol{\Lambda}_{jj}$ on the diagonal.

The diagonal entries of $\boldsymbol{\Lambda}$ are known as **singular values** of $\boldsymbol{X}$.

Theorem (Singular Value Decomposition) - II

Using the SVD we can write the following (remember that we have assumed that $\boldsymbol{X}$ is centred)

$$\boldsymbol{S} = \frac{1}{n-1}\boldsymbol{X}^T\boldsymbol{X} = \frac{1}{n-1}\boldsymbol{Q}\boldsymbol{\Lambda}^T\boldsymbol{P}^T\boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{Q}^T = \frac{1}{n-1}\boldsymbol{Q}\boldsymbol{\Lambda}^2\boldsymbol{Q}^T$$

where, when $n > p$, $\boldsymbol{\Lambda}^2$ is a $p \times p$ diagonal matrix with diagonal elements $\boldsymbol{\Lambda}^2_{jj}$.

Note that this has exactly the same form as the eigendecomposition of $\boldsymbol{S} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T$ where $\boldsymbol{V} = \boldsymbol{Q}$ and $\boldsymbol{D} = \frac{1}{n-1}\boldsymbol{\Lambda}^2$.

Thus the eigendecomposition of the covariance matrix can be recovered from the SVD of the (centred) data matrix.

Theorem (Singular Value Decomposition) - III

Therefore

$$\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{V} = \boldsymbol{X}\boldsymbol{Q} = \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{P}\boldsymbol{\Lambda}$$

which implies that we could obtain the scores matrix by calculating $\boldsymbol{P}$ and $\boldsymbol{\Lambda}$, and we do not necessarily need eigenvectors explicitly.

We could achieve this alternatively using the eigendecomposition of another matrix: $n \times n$ matrix $\boldsymbol{X}\boldsymbol{X}^T$ since

$$\boldsymbol{X}\boldsymbol{X}^T = \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{\Lambda}^T\boldsymbol{P}^T = \boldsymbol{P}\boldsymbol{\Lambda}^2\boldsymbol{P}^T$$

where, when $p > n$, $\boldsymbol{\Lambda}^2$ is here the $n \times n$ diagonal matrix $[\boldsymbol{\Lambda}_{ii}^2]$.

The Gram matrix

$\boldsymbol{X}\boldsymbol{X}^T$ is the Gram matrix. Every entry is an inner product between data vectors i.e.

$$[\boldsymbol{X}\boldsymbol{X}^T]_{ij} = \sum_{\ell=1}^{p} x_{i\ell} x_{j\ell} = x_i^T x_j$$

when $n \ll p$: Calculating the eigendecomposition of $\boldsymbol{X}\boldsymbol{X}^T$ scales like $O(n^3)$ which is much less than the eigendecomposition of $\boldsymbol{X}^T\boldsymbol{X}$ which scales like $O(p^3)$. Also, it is much easier to store and work with an $n \times n$ matrix than a $p \times p$ matrix.

In practice, the svd algorithm on the matrix $\boldsymbol{X}$ directly is more stable and programmed (in eg `R`) efficiently, and is used in the routine `prcomp`. When $p > n$ it returns the loading matrix $\boldsymbol{V}$ as a $p \times n$ matrix. The complexity is $O(np \min(n, p))$.

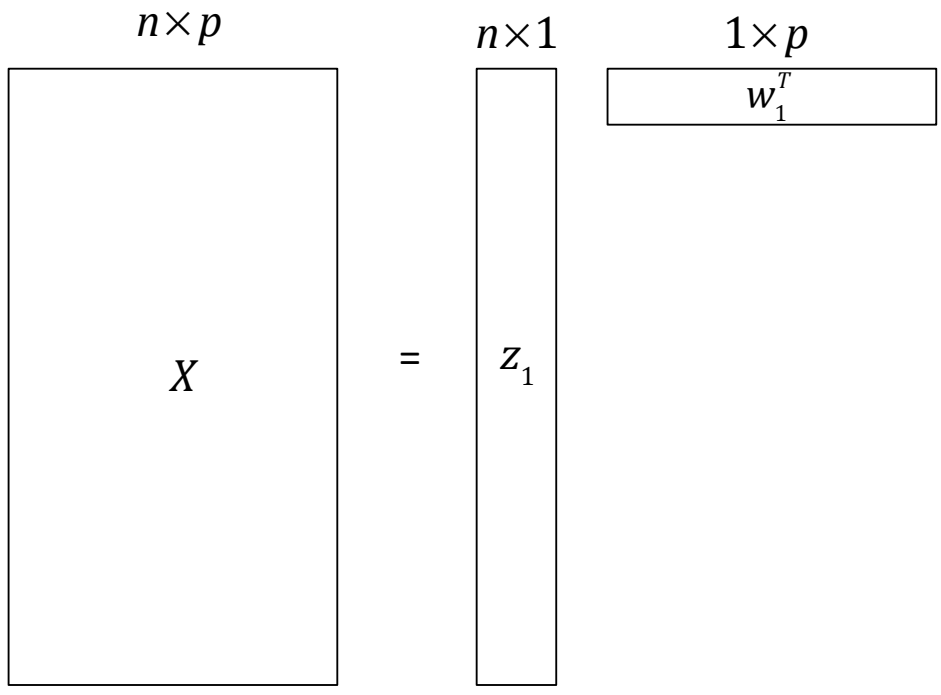
PCA as minimizing reconstruction error - I

A different view of PCA. If $\boldsymbol{X}$ is our $n \times p$ data matrix then a rank-1 approximation to the matrix can be constructed as

$$\widetilde{\boldsymbol{X}} = z_1 w_1^T$$

where z_1 is an n-column vector and w_1 is p-column vector. Together the product $z_1 w_1^T$ is an $n \times p$ matrix, and the idea is that we find the best pair of vectors z_1 and w_1 to minimize the distance.

Rank-1 approximation to the data matrix



PCA as minimizing reconstruction error - II

We can measure the distance between $\boldsymbol{X}$ and $\widetilde{\boldsymbol{X}}$ by sum of the squared differences between the two matrices, defined as $\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{p}(\boldsymbol{X}_{ij}-\widetilde{\boldsymbol{X}}_{ij})^2$.

The ith row of $\boldsymbol{X}$ is x_i^T. The ith row of $\widetilde{\boldsymbol{X}}$ is $z_{i1}w_1^T$.

$$
\begin{aligned}
J(z_1, w_1) &= \frac{1}{n}\sum_{i=1}^{n}(x_i - z_{i1}w_1)^T(x_i - z_{i1}w_1) \\
&= \frac{1}{n}\sum_{i=1}^{n}[x_i^T x_i - 2z_{i1}w_1^T x_i + w_1^T z_{i1}^T z_{i1} w_1] \\
&= \frac{1}{n}\sum_{i=1}^{n}[x_i^T x_i - 2z_{i1}w_1^T x_i + z_{i1}^2(w_1^T w_1)]
\end{aligned}
$$

PCA as minimizing reconstruction error - III

Since we can arbitrarily scale z_1 and w_1 so that the product $\widetilde{\boldsymbol{X}} = z_1 w_1^T$ stays the same, we can add the constraint that $w_1^T w_1 = 1$. Taking derivatives wrt z_{i1} and equating to zero gives

$$\frac{\partial}{\partial z_{i1}} J(z_1, w_1) = \frac{1}{n}[-2w_1^T x_i + 2z_{i1}] = 0 \Rightarrow z_{i1} = w_1^T x_i = x_i^T w_1$$

or in matrix form

$$z_1 = \boldsymbol{X} w_1$$

PCA as minimizing reconstruction error - IV

Plugging this back in gives

$$\begin{aligned} J(w_1) &= \frac{1}{n}\sum_{i=1}^{n} x_i^T x_i - \frac{1}{n}\sum_{i=1}^{n} z_{i1}^2 \\ &= \frac{1}{n}\sum_{i=1}^{n} x_i^T x_i - \frac{1}{n}\sum_{i=1}^{n} (w_1^T x_i)(w_1^T x_i)^T \\ &= constant - \frac{1}{n}\sum_{i=1}^{n} w_1^T x_i x_i^T w_1 \end{aligned}$$

Now if we assume that the columns of $\boldsymbol{X}$ have been mean centered then

$$\widehat{\boldsymbol{\Sigma}} = \frac{1}{n}\boldsymbol{X}^T\boldsymbol{X} = \frac{1}{n}\sum_{i=1}^{n} x_i x_i^T$$

which gives

$$J(w_1) = constant - w_1^T \widehat{\boldsymbol{\Sigma}} w_1 \tag{7}$$

PCA as minimizing reconstruction error - V

To minimize this we need to maximize $w_1^T \widehat{\mathbf{\Sigma}} w_1$ subject to the constraint $w_1^T w_1 = 1$. We can replace $\widehat{\mathbf{\Sigma}}$ with $\boldsymbol{S}$ since $\boldsymbol{S} \propto \widehat{\mathbf{\Sigma}}$ and solve using Lagrange multipliers

$$\max_{w_1} w_1^T \boldsymbol{S} w_1 \quad \text{subject to} \quad w_1^T w_1 = 1$$

which leads to the eigenvalue equation

$$\boldsymbol{S} w_1 = \lambda_1 w_1 \tag{8}$$

This is the same equation we had before (see equation (1)), so w_1 and λ_1 are an eigenvector and eigenvalue of $\boldsymbol{S}$ respectively. Since $w_1^T \boldsymbol{S} w_1 = \lambda_1$ is what we are trying to maximize we choose λ_1 to be the largest eigenvalue of $\boldsymbol{S}$ and w_1 its associated eigenvector. Thus the best rank-1 approximation to $\boldsymbol{X}$ is $z_1 w_1^T = \boldsymbol{X} w_1 w_1^T$.

What about a rank-2 approximation?

This approach can be extended to find the best rank-2 approximation as

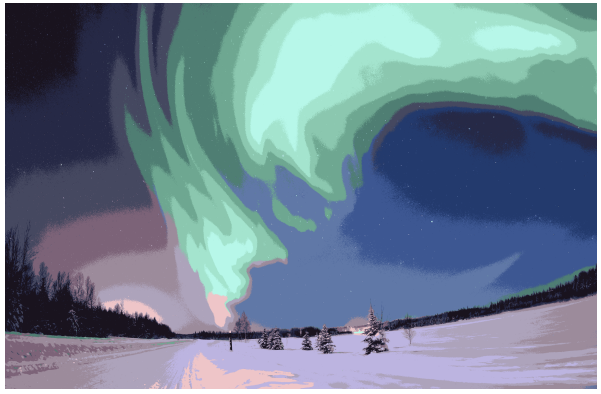
$$\boldsymbol{X} \approx z_1 w_1^T + z_2 w_2^T$$

using the constraints $w_2^T w_2 = 1$ and $w_2^T w_1 = 0$ and assuming that we have already estimated w_1 and z_1. It can be shown that w_2 is the eigenvector associated with the second largest eigenvalue of $\boldsymbol{S}$.
A general extension to the best rank-q approximation to $\boldsymbol{X}$ then follows from that, and can be written as

$$\boldsymbol{X} \approx \sum_{j=1}^{q} z_j w_j^T \tag{9}$$

Clustering

- Many datasets consist of multiple heterogeneous subsets.
- **Cluster analysis**: Given unlabelled data, we want algorithms that automatically group the datapoints into coherent subsets/clusters. Examples:
  - market segmentation of shoppers based on browsing and purchase histories
  - different types of cancer based on the gene expression measurements
  - discovering communities in social networks
  - image segmentation



The aim of clustering

- Clustering aims to group similar items together *and* to place separate dissimilar items into different groups
- Two objectives can contradict each other (similarity is not a transitive relation, while being in the same cluster is an equivalence relation)
- Notion of similarity/dissimilarity between data items is central: many ways to define and the choice will depend on the dataset being analyzed and dictated by domain specific knowledge
- *Partition-based* clustering: one divides n data items into K clusters $C_1, \ldots, C_K$ where for all $k, k' \in \{1, \ldots, K\}$,

$$C_k \subset \{1, \ldots, n\}\,, \quad C_k \cap C_{k'} = \emptyset \;\; \forall k \neq k', \quad \bigcup_{k=1}^{K} C_k = \{1, \ldots, n\}\,.$$

EU indicators dataset

Economic indicators for 27 EU countries (2011)

Country	CPI	UNE	INP	BOP	PRC	UN%
Belgium	116.03	4.77	125.59	908.60	6716.50	-1.60
Bulgaria	141.20	7.31	102.39	27.80	1094.70	3.50
CzechRep.	116.20	4.88	119.01	-277.90	2616.40	-0.60
Denmark	114.20	6.03	88.20	1156.40	7992.40	0.50
Germany	111.60	4.63	111.30	499.40	6774.60	-1.30
Estonia	135.08	9.71	111.50	153.40	2194.10	-7.70
Ireland	106.80	10.20	111.20	-166.50	6525.10	2.00
Greece	122.83	11.30	78.22	-764.10	5620.10	6.40
Spain	116.97	15.79	83.44	-280.80	4955.80	0.70
France	111.55	6.77	92.60	-337.10	6828.50	-0.90
Italy	115.00	5.05	87.80	-366.20	5996.60	-0.50
Cyprus	116.44	5.14	86.91	-1090.60	5310.30	-0.40
Latvia	144.47	12.11	110.39	42.30	1968.30	-3.60
Lithuania	135.08	11.47	114.50	-77.40	2130.60	-4.30
Luxembourg	118.19	3.14	85.51	2016.50	10051.60	-3.00
Hungary	134.66	6.77	115.10	156.20	1954.80	-0.10
Malta	117.65	4.15	101.65	359.40	3378.30	-0.60
Netherlands	111.17	3.23	103.80	1156.60	6046.00	-0.40
Austria	114.10	2.99	116.80	87.80	7045.50	-1.50
Poland	119.90	6.28	146.70	-74.80	2124.20	-1.00
Portugal	113.06	9.68	89.30	-613.40	4073.60	0.80
Romania	142.34	4.76	131.80	-128.70	1302.20	3.20
Slovenia	118.33	5.56	105.40	39.40	3528.30	1.80
Slovakia	117.17	9.19	156.30	16.00	2515.30	-2.10
Finland	114.60	5.92	101.00	-503.70	7198.80	-1.30
Sweden	112.71	6.10	100.50	1079.10	7476.70	-2.30
UnitedKingdom	120.90	6.11	90.36	-24.30	6843.90	-0.80
Variance	111.66	9.95	357.27	450057.15	5992520.48	7.12

CPI=consumer price index; UNE=unemployment; INP=industrial production; BOP=balance of payments; PRC=private final consumption expenditure; UN%=annual change in unemployment rate; source= EUROSTAT

Within-cluster deviance

Goal: divide data items into a *pre-assigned number K of clusters* $C_1, \ldots, C_K$ where for all $k, k' \in \{1, \ldots, K\}$,

$$C_k \subset \{1, \ldots, n\}\,, \quad C_k \cap C_{k'} = \emptyset \;\; \forall k \neq k', \quad \bigcup_{k=1}^{K} C_k = \{1, \ldots, n\}\,.$$

Define $W(C_k)$ to be a measure of how different the observations are within cluster k, the most common choice is to use squared distances:

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \|x_i - x_{i'}\|_2^2 = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2$$

Problem sheet:

$$\frac{1}{|C_k|} \sum_{i,i' \in C_k} \|x_i - x_{i'}\|_2^2 = 2 \sum_{i \in C_k} \|x_i - \mu_k\|_2^2, \tag{10}$$

where $\mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i$ is the mean of the observations in cluster k.

Within-cluster deviance

Each cluster is represented using a *prototype* or *cluster centroid* μ_k.

Within-cluster deviance:

$$W(C_k, \mu_k) = \sum_{i \in C_k} \|x_i - \mu_k\|_2^2 = \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \mu_{kj})^2$$

The overall quality of the clustering is given by the total within-cluster deviance:

$$W = \sum_{k=1}^{K} W(C_k, \mu_k) = \sum_{k=1}^{K} \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \mu_{kj})^2$$

K-means

$$W = \sum_{k=1}^{K} \sum_{i \in C_k} \|x_i - \mu_k\|_2^2 = \sum_{i=1}^{n} \|x_i - \mu_{c_i}\|_2^2$$

where $c_i = k$ if and only if $i \in C_k$.

- Given partition $\{C_k\}$, we can find the optimal prototypes easily by differentiating W with respect to μ_k:

$$\frac{\partial W}{\partial \mu_k} = 2 \sum_{i \in C_k} (x_i - \mu_k) = 0 \qquad \Rightarrow \mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i$$

- Given prototypes, we can easily find the optimal partition by assigning each data point to the closest cluster prototype:

$$c_i = \text{argmin}_k \left\|x_i - \mu_k\right\|_2^2$$

But joint minimization over both is computationally difficult.

K-means

The K-means algorithm returns a *local optimum* of the objective function W, using iterative and alternating minimization.

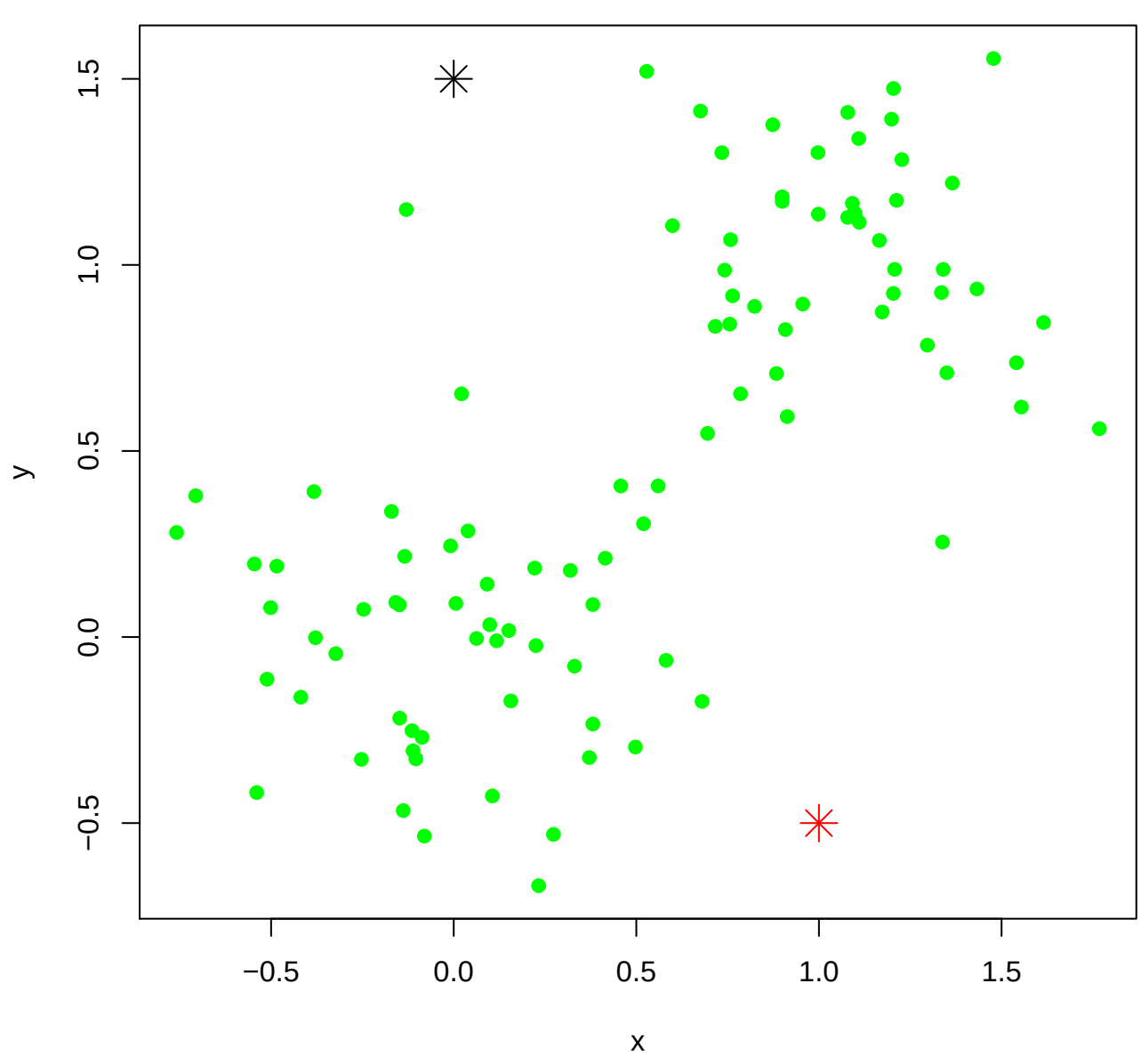
1. Randomly initialize K cluster centroids $\mu_1, \ldots, \mu_K$.
2. *Cluster assignment:* For each $i = 1, \ldots, n$, assign each x_i to the cluster with the nearest centroid,

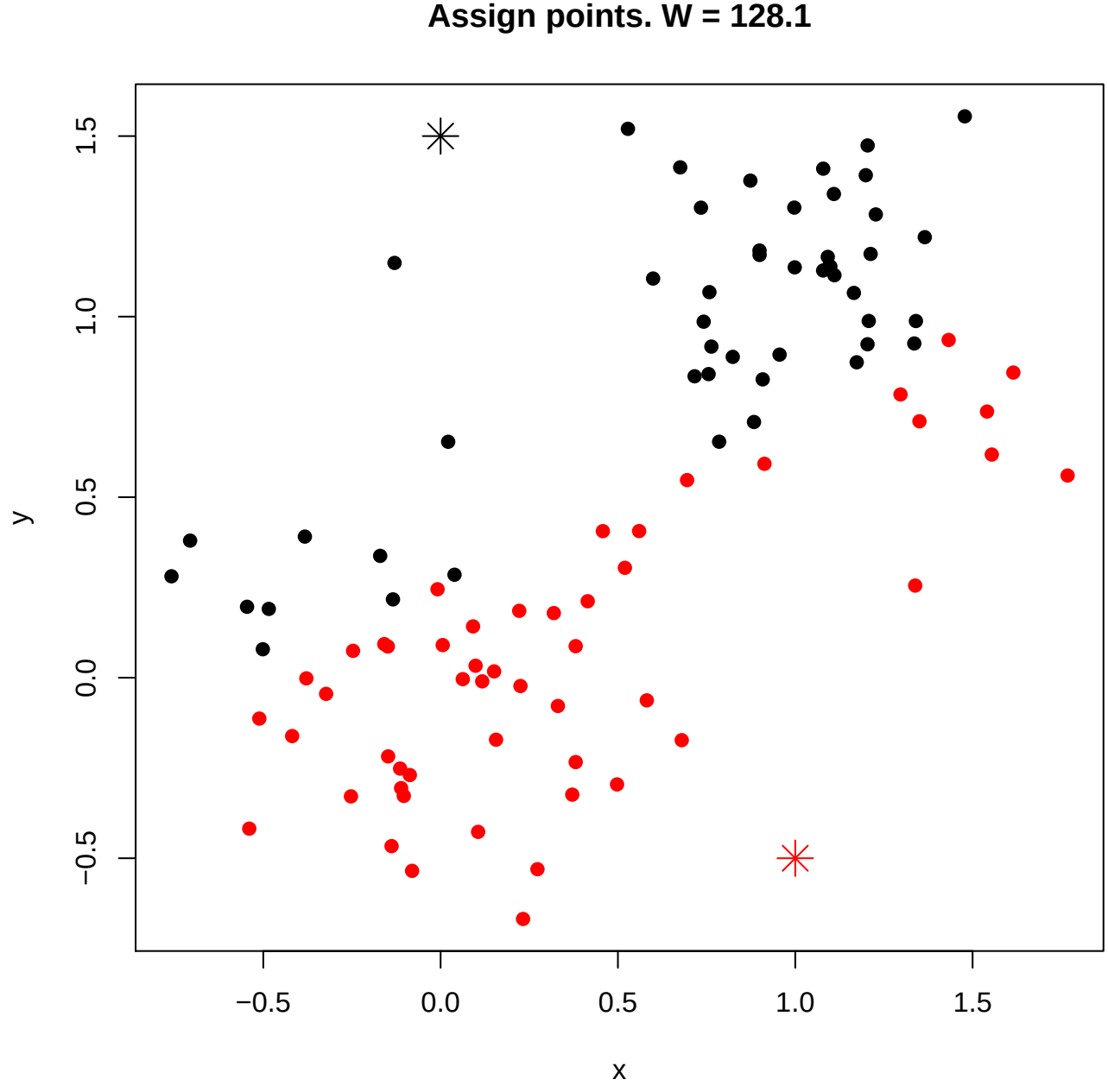
$$c_i := \text{argmin}_k \left\| x_i - \mu_k \right\|_2^2$$

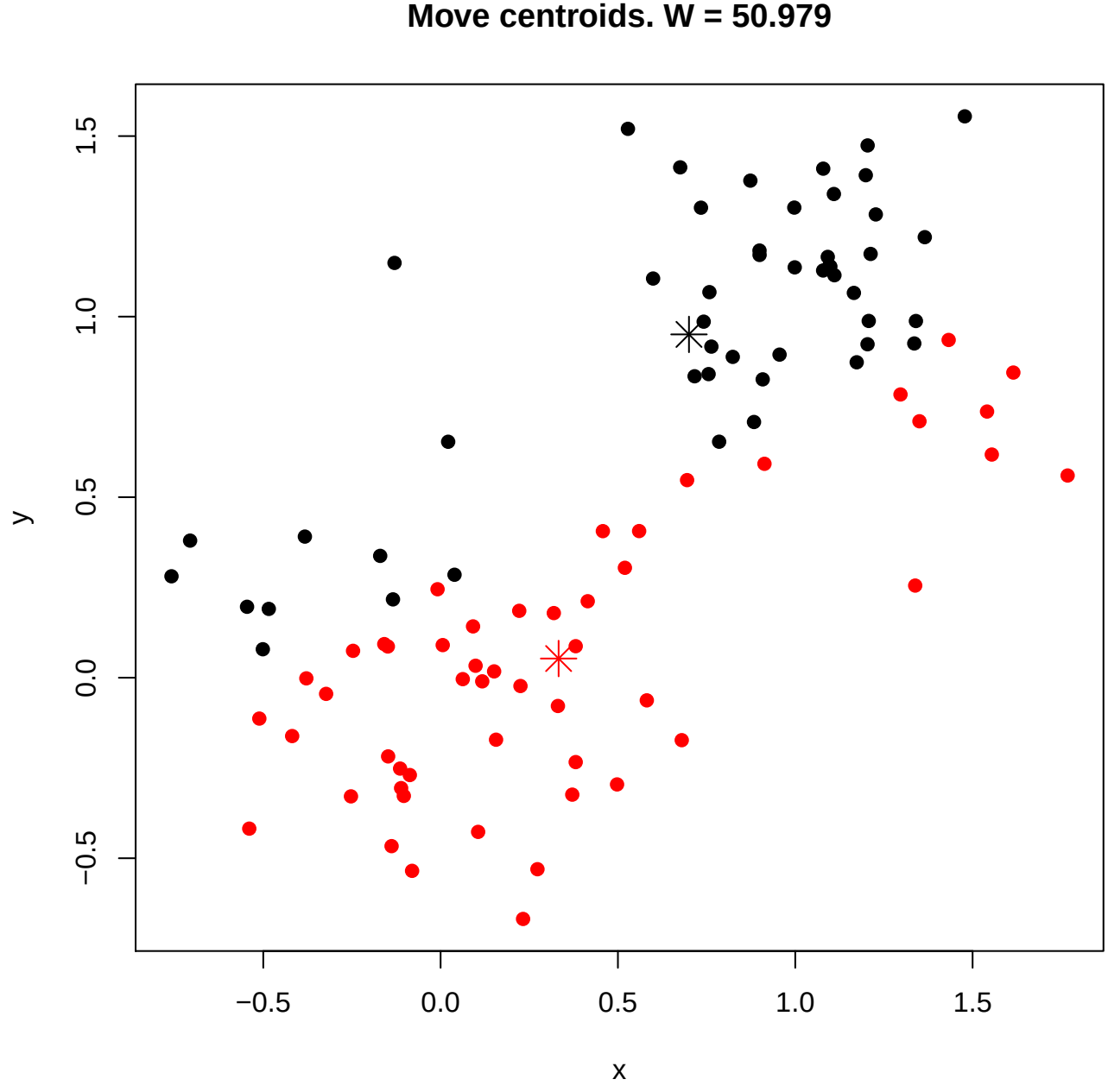
   Set $C_k := \{i : c_i = k\}$ for each k.
3. *Move centroids:* Set $\mu_1, \ldots, \mu_K$ to the averages of the new clusters:

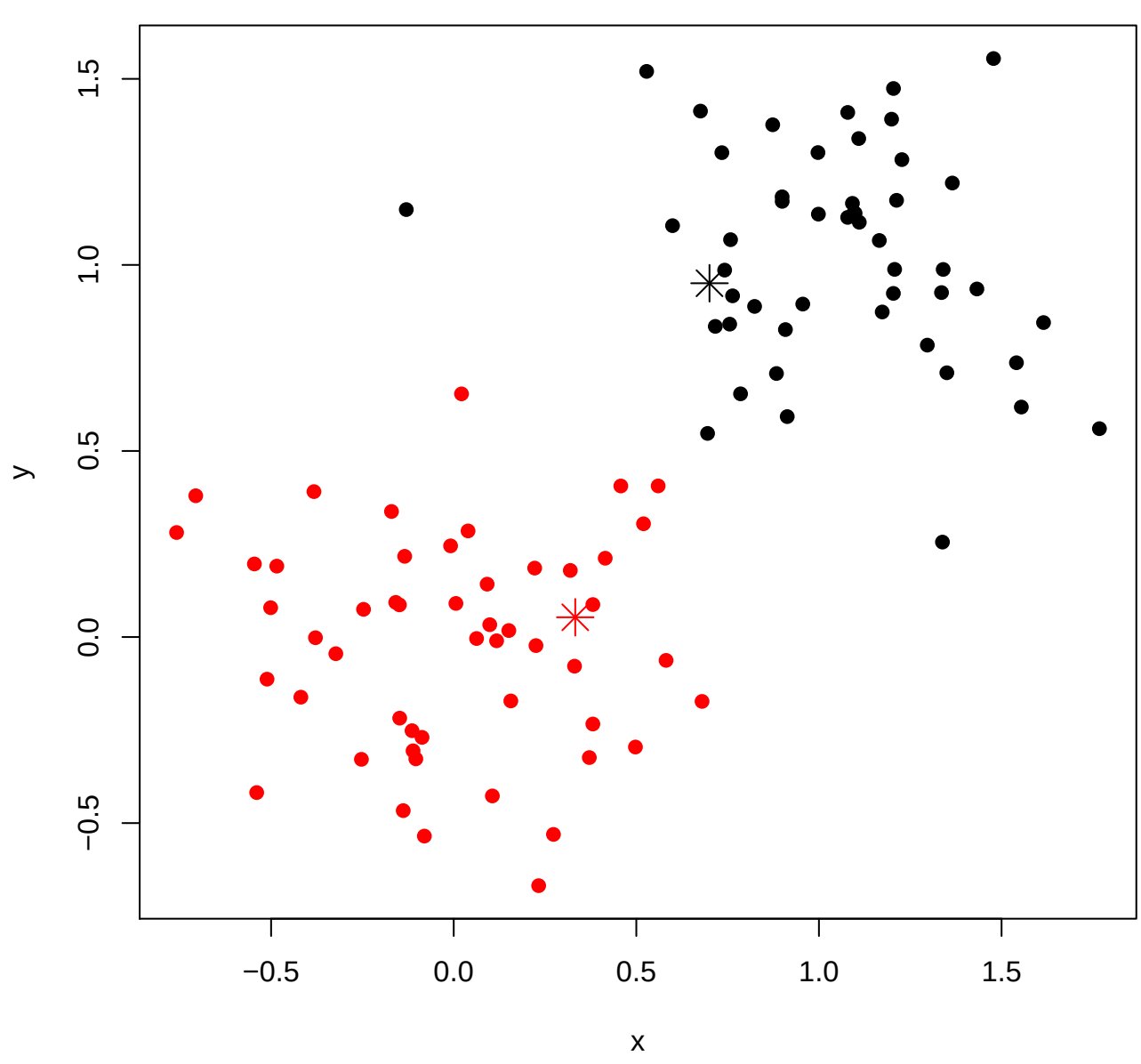
$$\mu_k := \frac{1}{|C_k|} \sum_{i \in C_k} x_i$$

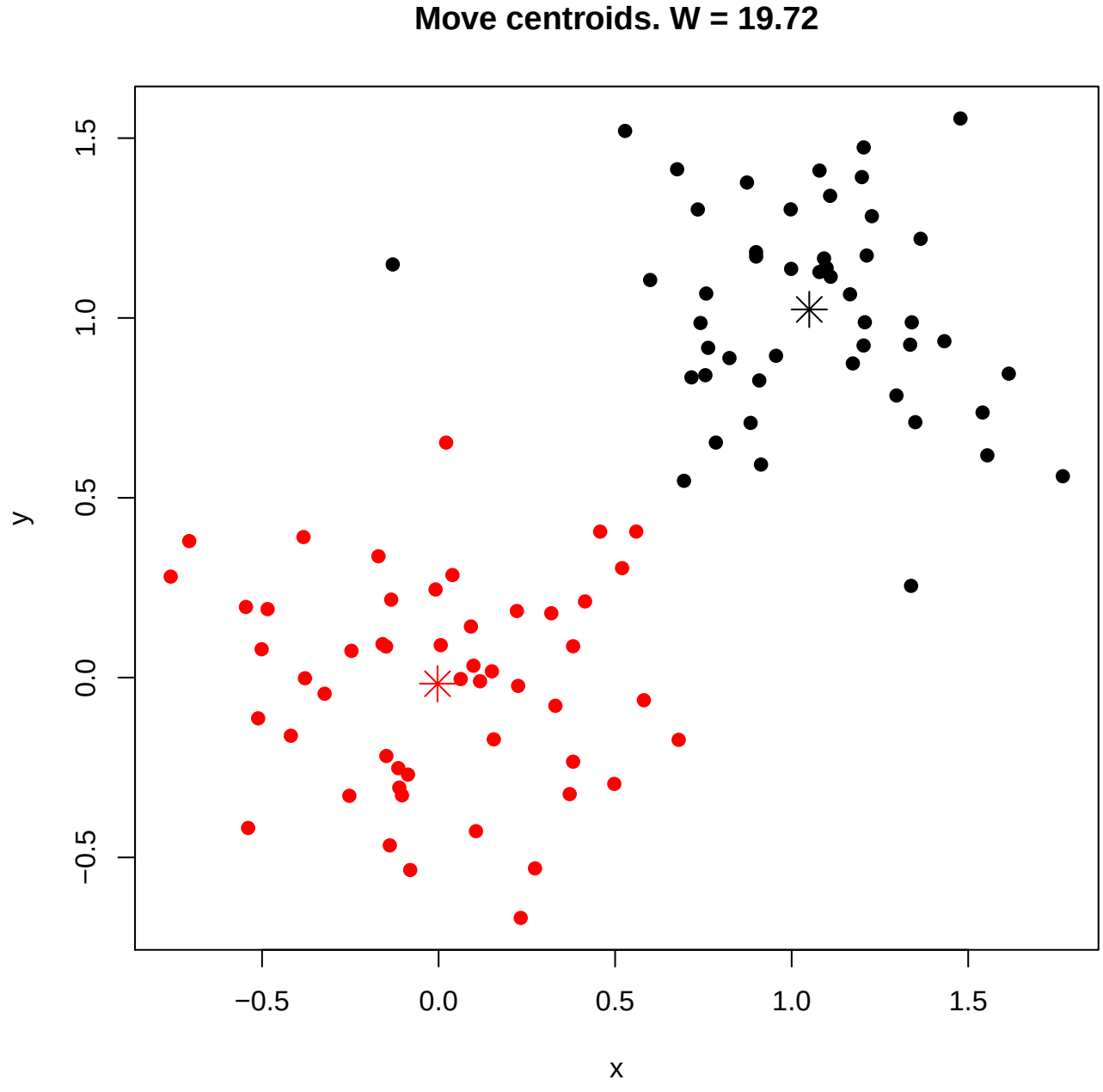
4. Repeat steps 2-3 until convergence.
5. Return the partition $\{C_1, \ldots, C_K\}$ and means $\mu_1, \ldots, \mu_K$.

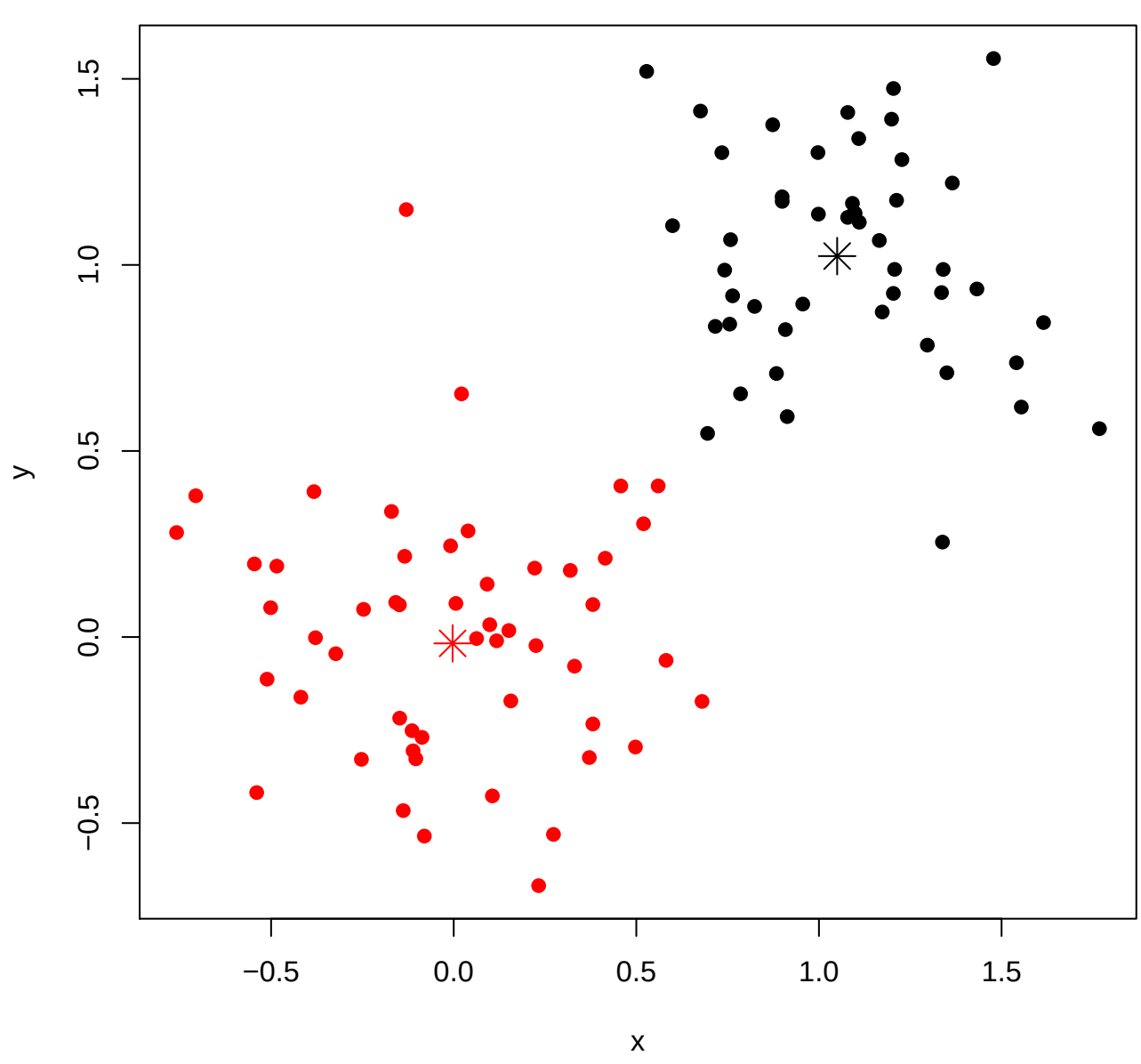

K−means illustration
y
x
1.5
1.0
0.5
0.0
−0.5
−0.5
0.0
0.5
1.0
1.5

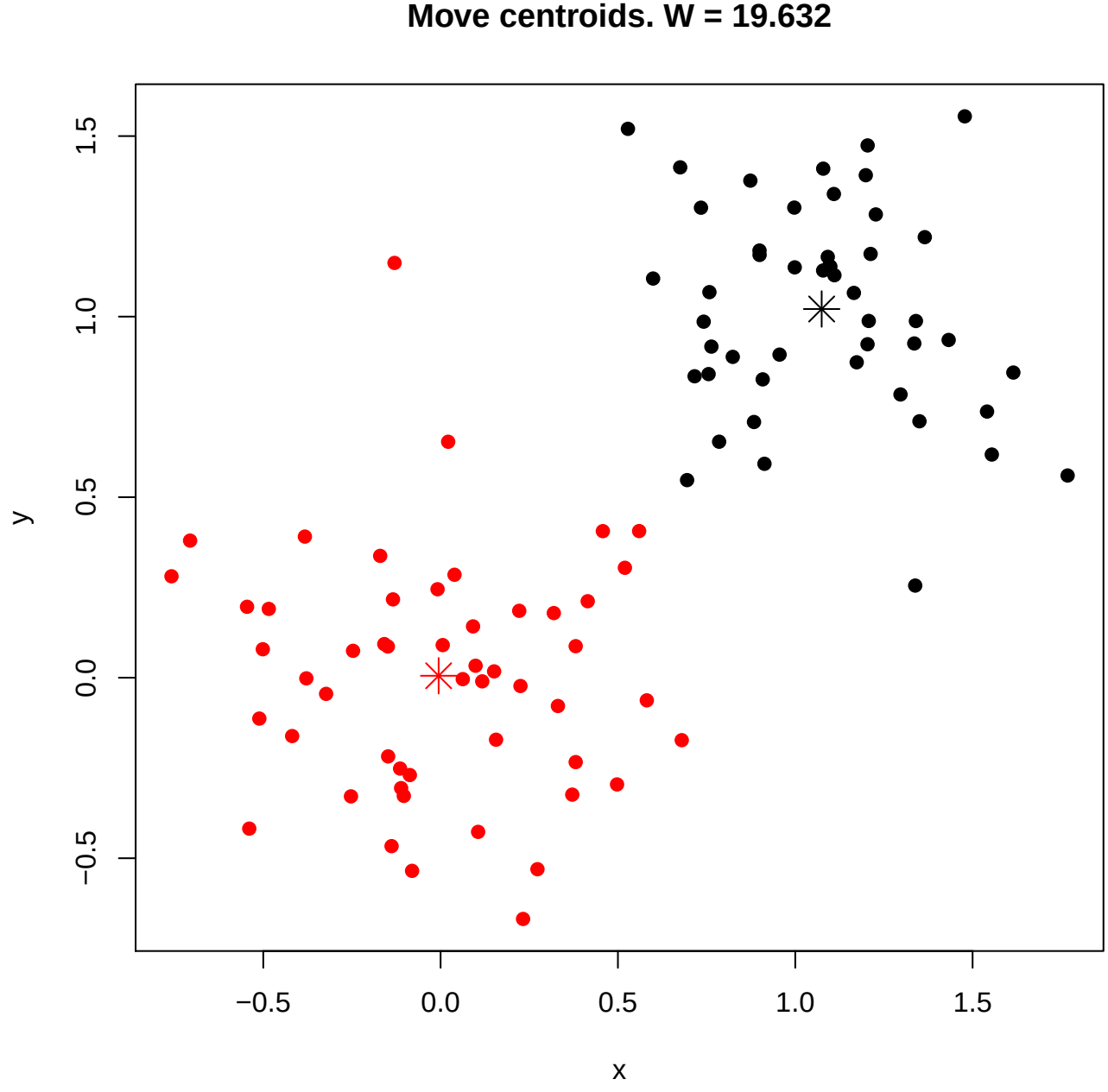

Assign points. W = 128.1
y
x
1.5
1.0
0.5
0.0
−0.5
−0.5
0.0
0.5
1.0
1.5


Move centroids. W = 50.979
y
x
1.5
1.0
0.5
0.0
−0.5
−0.5
0.0
0.5
1.0
1.5


Assign points. W = 31.969
y
x
1.5
1.0
0.5
0.0
−0.5
−0.5
0.0
0.5
1.0
1.5

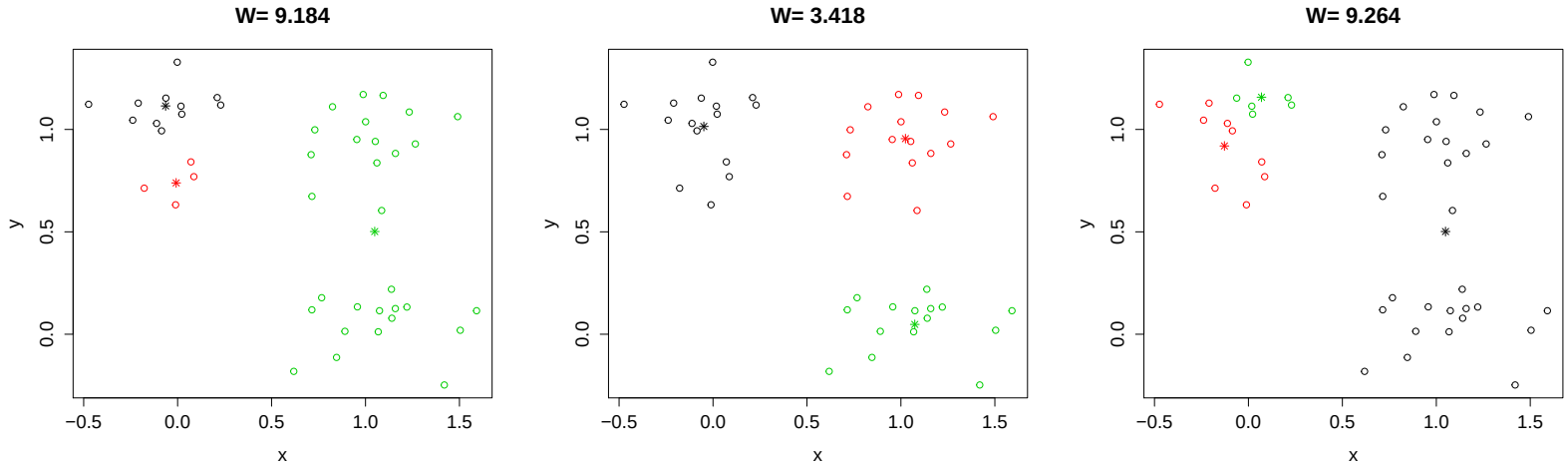

Move centroids. W = 19.72
y
x
1.5
1.0
0.5
0.0
−0.5
−0.5
0.0
0.5
1.0
1.5


Assign points. W = 19.688
y
x
1.5
1.0
0.5
0.0
−0.5
−0.5
0.0
0.5
1.0
1.5

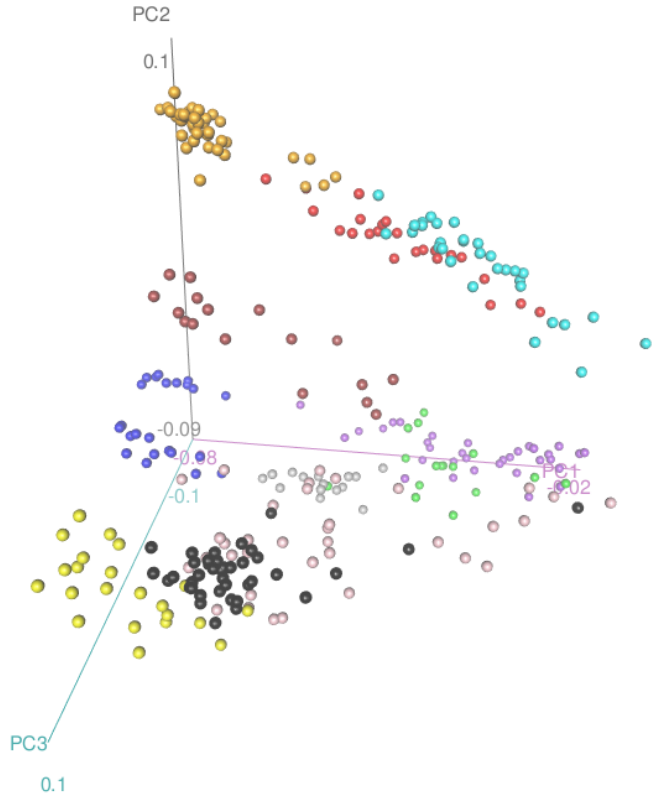

Move centroids. W = 19.632
y
x
1.5
1.0
0.5
0.0
−0.5
−0.5
0.0
0.5
1.0
1.5

K-means

- *The algorithm stops in a finite number of iterations.* Between steps 2 and 3, W either stays constant or it decreases, this implies that we never revisit the same partition. As there are only finitely many partitions, the number of iterations cannot exceed this.
- *The K-means algorithm need not converge to global optimum.* K-means can get stuck at suboptimal configurations and the result depends on the starting configuration. Typically perform a number of runs from different initial values, and pick the end result with minimum W.



K-means clustering - single cell dataset



Visit movie.gif here
`https://courses.maths.ox.ac.uk/course/view.php?id=6033`
for 3D PCA projection.

Convergence of the K-means algorithm

It can be shown (see Exercises) that for a cluster C_k

$$\frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 = 2 \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \mu_{kj})^2 \tag{11}$$

so the problem can be re-written as

$$\min_{C_1, C_2, \ldots, C_k} \left\{ \sum_{k=1}^{K} \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \mu_{kj})^2 \right\} \tag{12}$$

The objective function

We call

$$\sum_{k=1}^{K} \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \mu_{kj})^2 \tag{13}$$

the **objective function** (total within-cluster deviance) that we are trying to minimize.

Also, for any set of p-vectors $v_1, \ldots, v_K$ it can be shown (see Exercises) that

$$\sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - v_{kj})^2 = \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \mu_{kj})^2 + |C_k| \sum_{j=1}^{p} (\mu_{kj} - v_{kj})^2 \tag{14}$$

Let $C^{(t)}$ denote the clustering at iteration t.

Let $\mu_k^{(t)}$ be the mean of observations in cluster k at iteration t.

Now suppose that we have just updated the cluster assignments for the next iteration i.e we have determined what $C^{(t+1)}$ is and so the current value of the objective function (using $C^{(t+1)}$ and $\mu_k^{(t)}$) is

$$\sum_{k=1}^{K} \sum_{i \in C_k^{(t+1)}} \sum_{j=1}^{p} (x_{ij} - \mu_{kj}^{(t)})^2$$

But since $\mu_k^{(t)}$ may not be the same as the cluster means for the assignments $C^{(t+1)}$ (since the cluster assignments can change) when we update the cluster means to $\mu_k^{(t+1)}$ we must have that

$$\sum_{k=1}^{K} \sum_{i \in C_k^{(t+1)}} \sum_{j=1}^{p} (x_{ij} - \mu_{kj}^{(t+1)})^2 \leq \sum_{k=1}^{K} \sum_{i \in C_k^{(t+1)}} \sum_{j=1}^{p} (x_{ij} - \mu_{kj}^{(t)})^2$$

So updating the cluster means to $\mu_k^{(t+1)}$ never increases the objective function.

Similarly, after this step the cluster means are fixed and we update the assignments.

We can re-write the objective function as a sum over the n observations

$$\sum_{i=1}^{n}\sum_{j=1}^{p}(x_{ij}-\mu_{c_i j})^2.$$

This is clearly minimized by assigning each observation to the cluster with minimum Euclidean distance to the cluster mean.

Multiple starts

The algorithm does not always give the same solution since the start point is random.

https://shabal.in/visuals/kmeans/1.html

Choosing the number of clusters - I

The choice of K is a key first step in the algorithm.

What is the best choice of K?

This is an example of **model choice**

Wrong approach : choose K by fitting the model to the dataset for many different values of K and then picking the value that results in a minimum of the objective. This will always occur when $K = n$ and assigning each observation to its own cluster.

Choosing the number of clusters - II

What's wrong with having so many clusters?

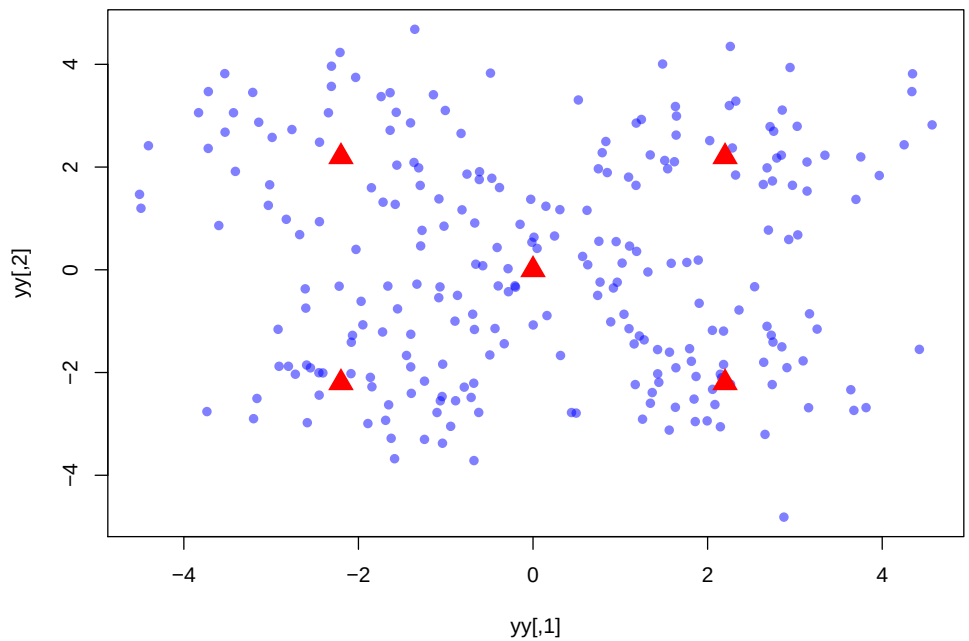
This is almost never a realistic or meaningful solution to the problem.

This is an example of **over-fitting**, in which we use a model that has too many parameters (in case the $K = n$ cluster means) relative to the number of observations.

Better solutions involve working with an objective function that penalizes models with an increasing number of parameters. Such approaches to model choice will be covered in later courses on Statistics and Data Analysis.

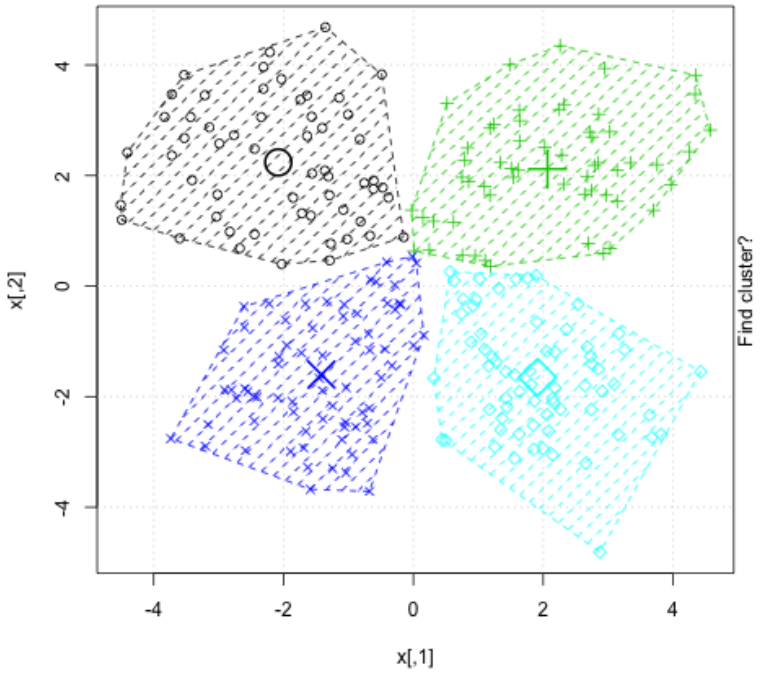
Disappearing Clusters?

Note that the solution could sometimes involve points allocated to less then K clusters. This happens when a cluster becomes empty during the run as no points are closest to its centroid.

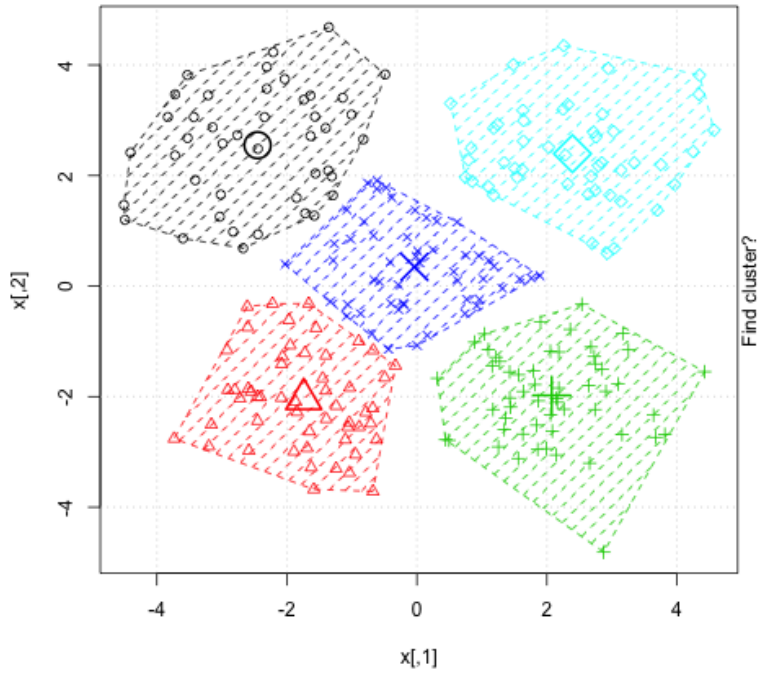


Example dataset with 5 close clusters of points.

Disappearing Clusters?



(a) Solution 1



(b) Solution 2

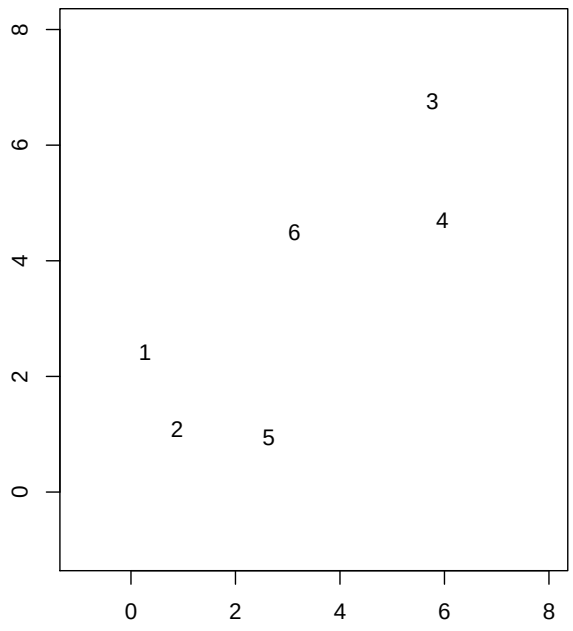
Hierarchical clustering

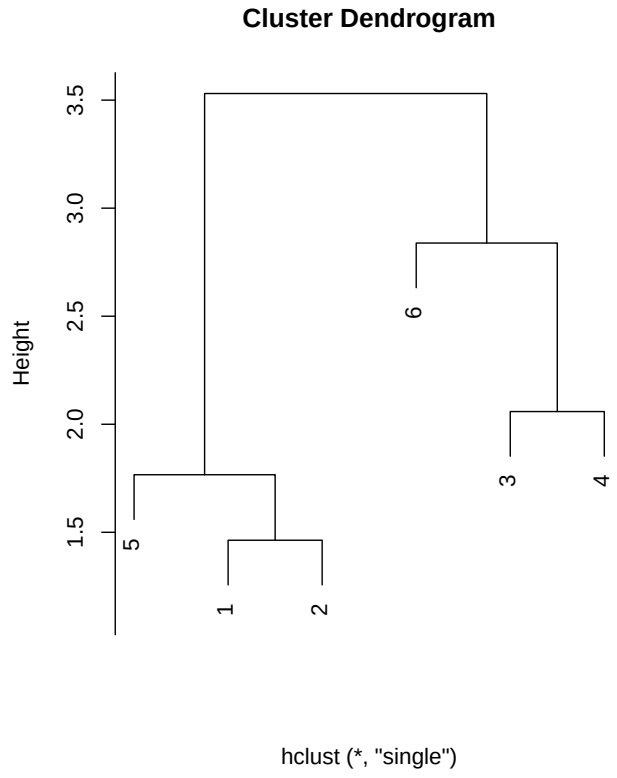
Hierarchical clustering avoids having to specify the number of clusters in advance.

The method results in a tree-like (hierarchical) representation of the dataset, that can be helpful when visualising structure in the dataset.

In this course we will describe **agglomerative clustering** approaches. The methods are called agglomerative because they iteratively fuse pairs observations together to form clusters.

Hierarchical clustering - example 1





Quantifying dissimilarities

Begin with n observations and a measure of all the $\binom{n}{2} = n(n-1)/2$ pairwise dissimilarities, denoted $d_{ii'}$ for $i \neq i' \in (1, \ldots, n)$.

These dissimilarities can be represented in a lower diagonal matrix, denoted $\boldsymbol{D}^{(n)}$.

$$\boldsymbol{D}^{(n)} = \begin{bmatrix} d_{21} & & & & & \\ d_{31} & d_{32} & & & & \\ \vdots & \vdots & \ddots & & & \\ d_{i1} & d_{i2} & \ldots & d_{i(i-1)} & & \\ \vdots & \vdots & \ddots & \vdots & \ddots & \\ d_{n1} & d_{n2} & \ldots & d_{n(i-1)} & \ldots & d_{n(n-1)} \end{bmatrix}$$

Quantifying dissimilarities

It is common to use Euclidean distance to measure dissimilarity, ie

$$d_{ii'} = \|x_i - x_{i'}\|_2^2 = \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2,$$

but other options exist.

Iteratively fusing pairs

Start with treating each observation as its own cluster. Then,

For $i = n, n-1, \ldots, 2$

1. Find the pair of clusters with the smallest dissimilarity. Fuse these two clusters.
2. Compute the new dissimilarity matrix between the new fused cluster and all other $i-1$ remaining clusters and create an updated matrix of dissimilarities $\boldsymbol{D}^{(n-1)}$.

Linkage methods

A key step in the algorithm is the choice of how to compute the new dissimilarity (or distance) between the new fused cluster and all other $i - 1$ remaining clusters.

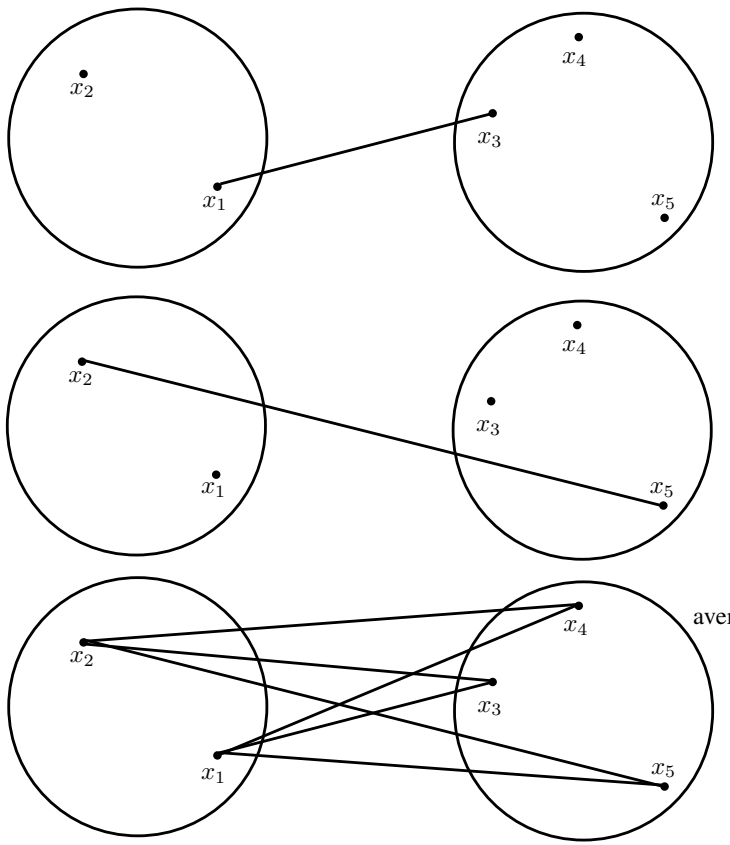
The are several commonly used options, with differing properties in terms of type of clusters they tend to produce.

Let G and H be two groups of observations then we can define the function $d(G, H)$ between groups G and H to be some function of all the pairwise dissimilarities $d_{ii'}$ where $i \in G$ and $i' \in H$.

Agglomerative Clustering

Iteratively join pairs of observations together to form clusters.

To join clusters C_i and $C_{i'}$ into larger clusters, we need a way to measure the dissimilarity $D(C_i, C_{i'})$ between them.



single linkage: $d(x_1, x_3)$

complete linkage: $d(x_2, x_5)$

average linkage: $\frac{1}{6}\sum_{i=1}^{2}\sum_{i'=3}^{5} d(x_i, x_{i'})$

Measuring Dissimilarity Between Clusters

To join clusters C_i and $C_{i'}$ into super-clusters, we need a way to measure the dissimilarity $D(C_i, C_{i'})$ between them.

(a) *Single Linkage*: elongated, loosely connected clusters

$$D(C_i, C_{i'}) = \min_{x,y} \left(d(x, y) | x \in C_i, y \in C_{i'} \right)$$

(b) *Complete Linkage*: compact clusters, relatively similar objects can remain separated at high levels

$$D(C_i, C_{i'}) = \max_{x,y} \left(d(x, y) | x \in C_i, y \in C_{i'} \right)$$

(c) *Average Linkage*: tries to balance the two above, but affected by the scale of dissimilarities

$$D(C_i, C_{i'}) = \text{avg}_{x,y} \left(d(x, y) | x \in C_i, y \in C_{i'} \right)$$

A small example

Consider the following small dataset with $n = 6$ and $p = 2$ variables

$$X = \begin{bmatrix} 0.27 & 2.42 \\ 0.88 & 1.09 \\ 5.77 & 6.76 \\ 5.96 & 4.71 \\ 2.64 & 0.94 \\ 3.13 & 4.49 \end{bmatrix}.$$

A small example: Dissimilarities between the 6 observations

Using Euclidean distance to measure dissimilarities results in the following matrix of dissimilarities, $d_{ii'}$ is the (i, i')th entry of the matrix.

$$\boldsymbol{D}^{(6)} = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 2 & \textbf{1.46} & & & & \\ 3 & 7.01 & 7.49 & & & \\ 4 & 6.13 & 6.24 & 2.06 & & \\ 5 & 2.79 & 1.77 & 6.61 & 5.02 & \\ 6 & 3.53 & 4.08 & 3.48 & 2.84 & 3.59 \end{array}$$

Note : we only show the lower triangle of the matrix and only record values to 2dp.

We use the notation $\boldsymbol{D}^{(k)}$ to denote the dissimilarities of k clusters of observations. The smallest entry is coloured in **red**.

A small example: Merging the two most similar observations

We will apply single linkage clustering to this dataset.

The smallest entry in the matrix is d_{21}, so we merge observations 1 and 2 into a new cluster (denoted (1,2)) and compute the new dissimilarities

$$\boldsymbol{D}^{(5)} = \begin{array}{c|cccc} & (1,2) & 3 & 4 & 5 \\ \hline 3 & 7.01 & & & \\ 4 & 6.13 & 2.06 & & \\ 5 & \mathbf{1.77} & 6.61 & 5.02 & \\ 6 & 3.53 & 3.48 & 2.84 & 3.59 \end{array}$$

A small example: The next steps

Next we merge clusters (1,2) and 5 into a new cluster (denoted (1,2,5)) and compute the new dissimilarities

$$D^{(4)} = \begin{array}{c|ccc} & (1,2,5) & 3 & 4 \\ \hline 3 & 6.61 & & \\ 4 & 5.02 & \mathbf{2.06} & \\ 6 & 3.53 & 3.48 & 2.84 \end{array}$$

A small example: The next steps

Next we merge clusters 3 and 4 into a new cluster (denoted (3,4)) and compute the new dissimilarities

$$\boldsymbol{D}^{(3)} = \begin{array}{c|cc} & (1,2,5) & (3,4) \\ \hline (3,4) & 5.02 & \\ 6 & 3.53 & \mathbf{2.84} \end{array}$$

A small example: The next steps

Next we merge clusters (3,4) and 6 into a new cluster (denoted (3,4,6) and compute the new dissimilarities

$$D^{(2)} = \begin{array}{c|c} & (1,2,5) \\ \hline (3,4,6) & \mathbf{3.53} \end{array}$$

Finally, we merge the two remaining clusters into a single cluster containing all the observations.

Notice how the sequence of dissimilarities which dictate which clusters merge (i.e. the numbers in red) are an increasing sequence of values.

Dendrograms

The results of an agglomerative clustering of a dataset can be represented as dendrogram, which is a tree-like diagram that allows us to visualise the way in which the observations have been joined into clusters.

A dendrogram is read from bottom to top. At the bottom we have a set of *leaves*. Each *leaf* of the dendrogram represents a data point. As we move up the tree leaves fuse into branches.

Dendrograms

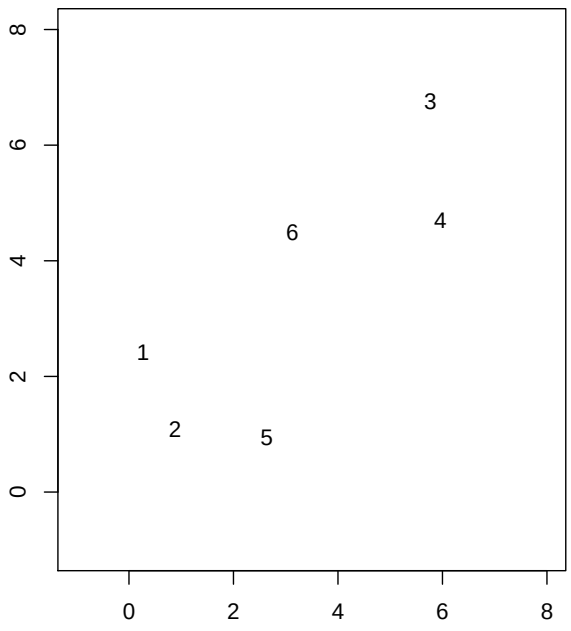
The first two leaves to merge are the two observations that merge into a cluster in the first step of the algorithm.

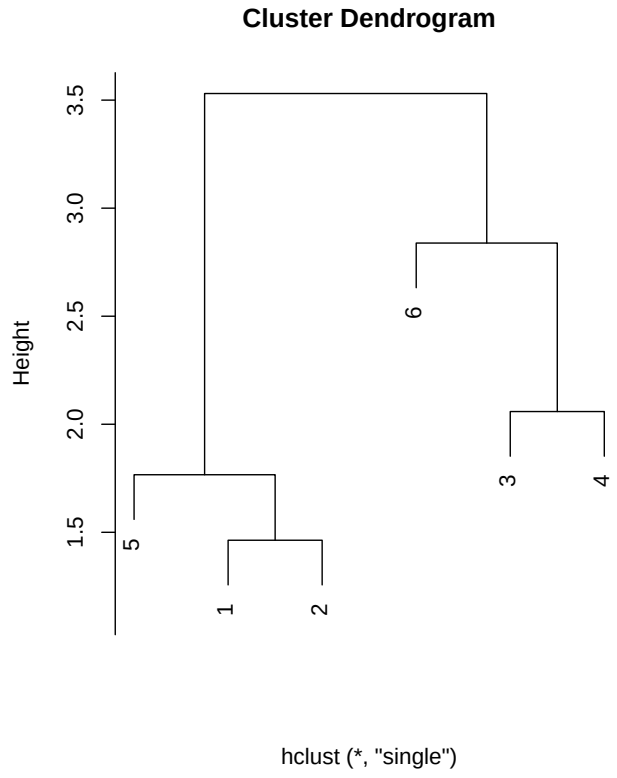
This merge point occurs on the y-axis at a value which is the dissimilarity between the two merging clusters.

The next two leaves to merge are the two observations that merge into a cluster in the second step of the algorithm, and so on.

The next slide shows the resulting dendrogram from the mini-example of 6 observations described above.

Hierarchical clustering - example 1





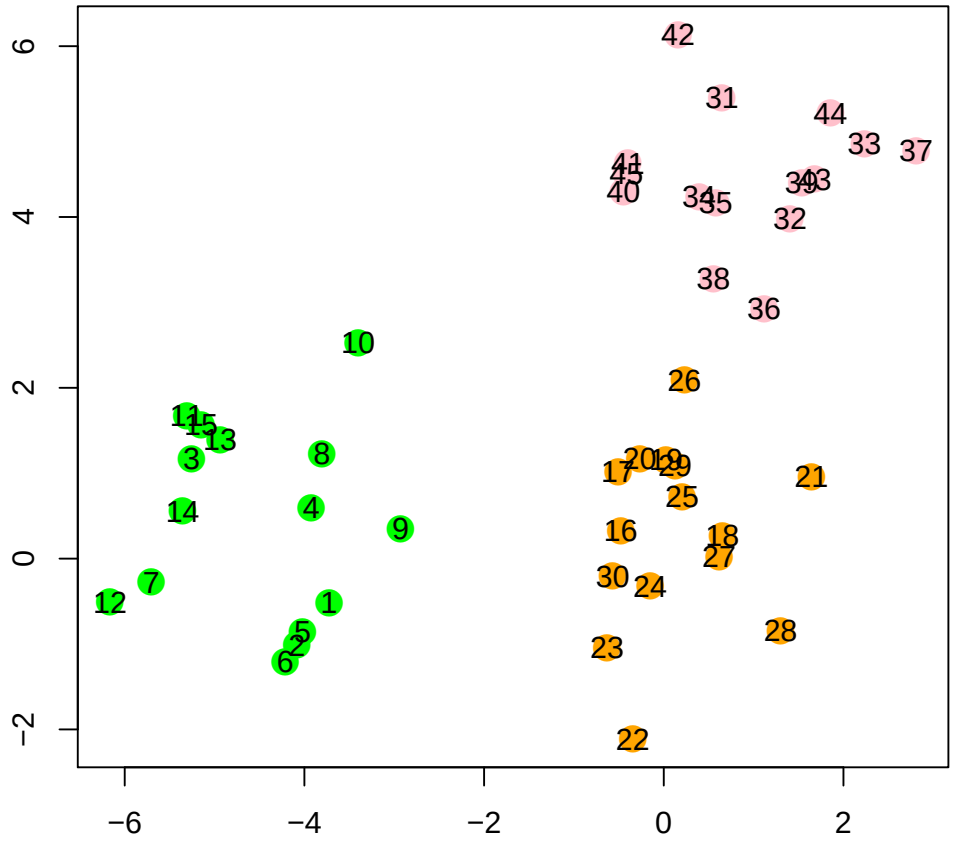
Differences between Linkage methods

Different linkage methods can lead to different dendrograms.

The next slide shows a new dataset of 45 points, with 15 points each simulated from 3 different bivariate normal distributions with different means.

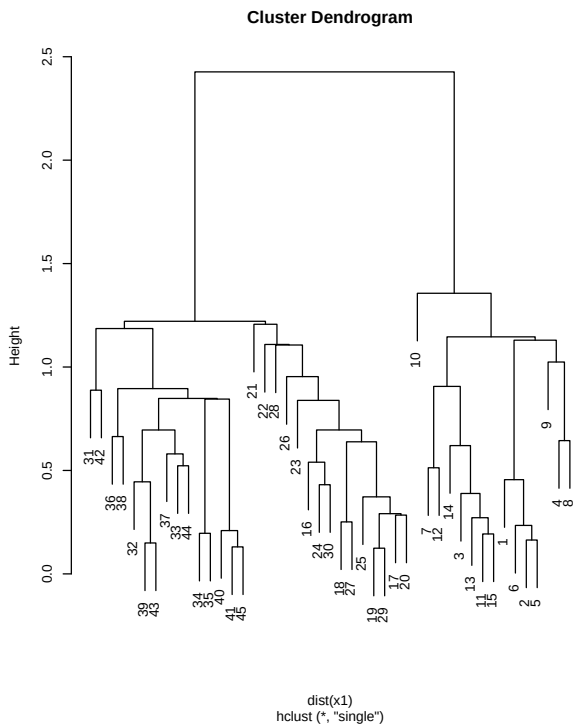
The 3 sets of points are coloured according to their true cluster. Points are also labelled with a number.

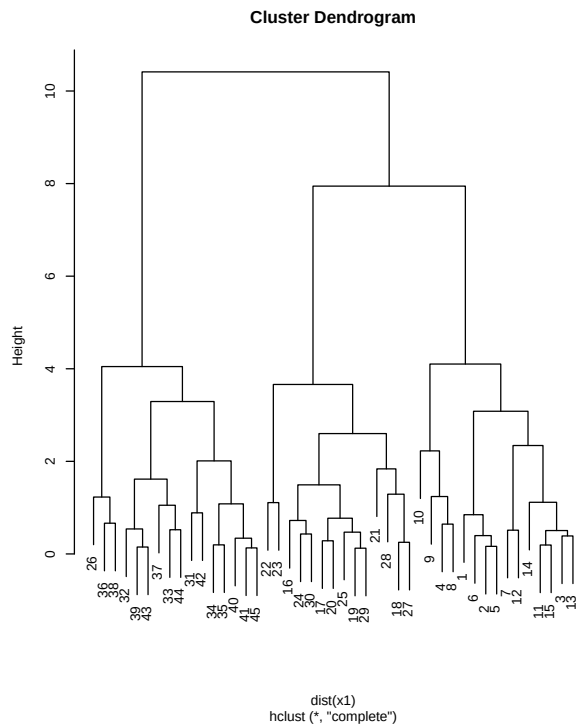
Hierarchical clustering - example 2

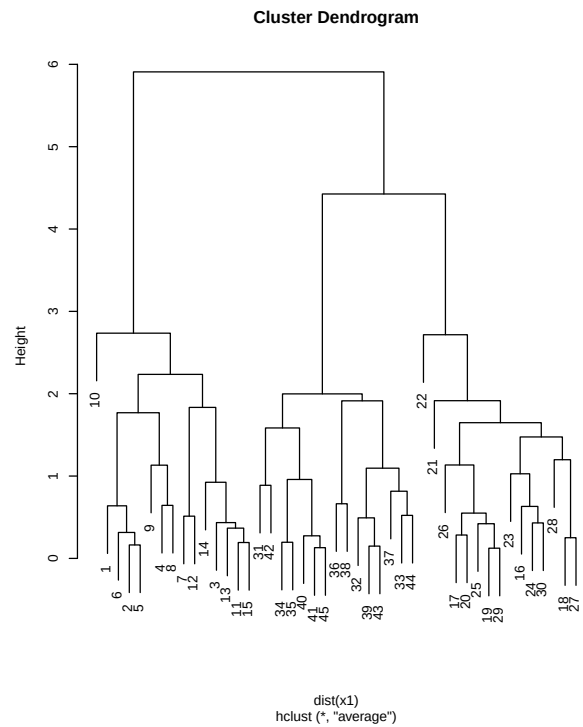


Hierarchical clustering - example 2

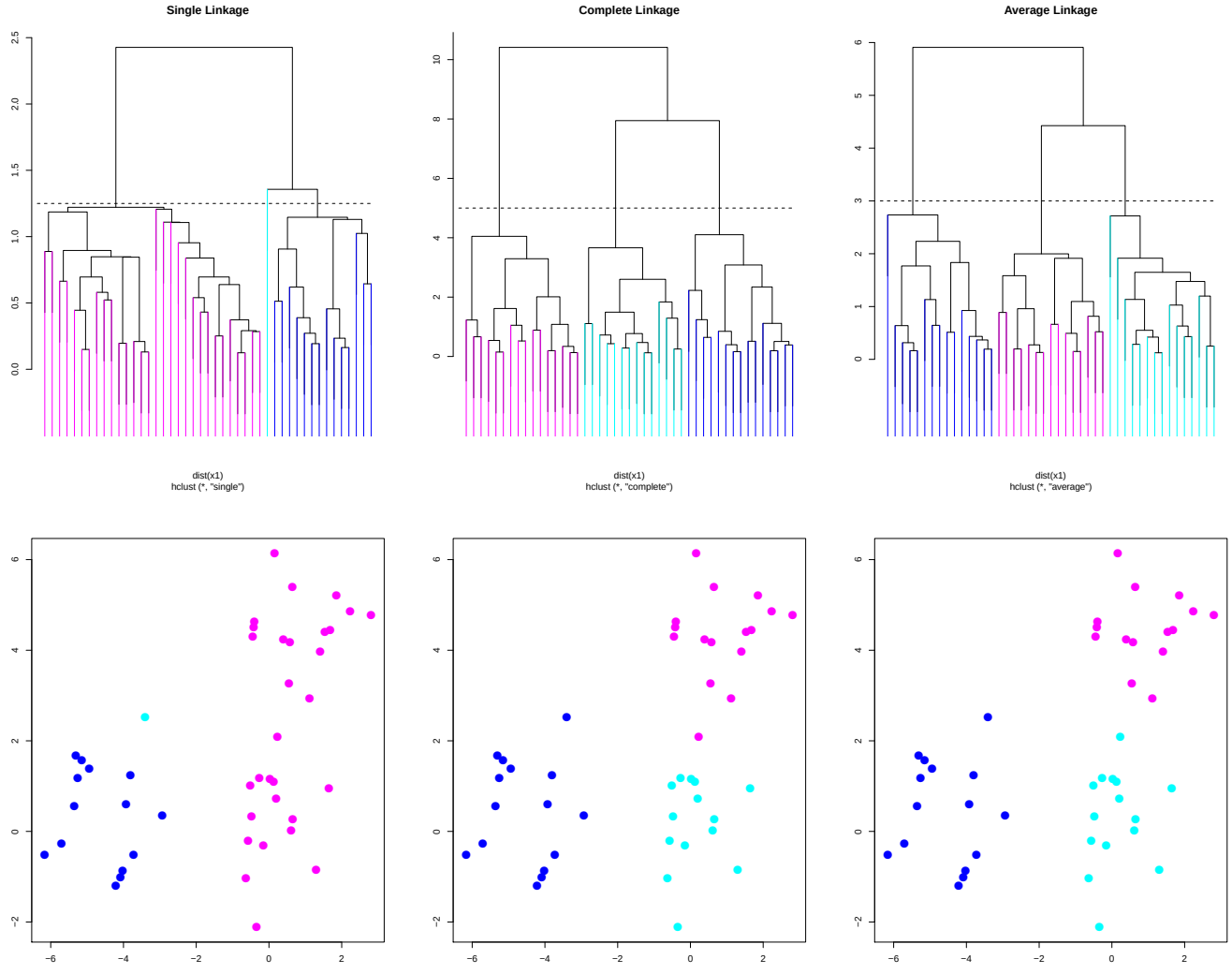
This figure hows the results of building dendrograms using Single Linkage, Complete Linkage and Average Linkage, and illustrates how these methods can differ.







Hierarchical clustering - extracting clusters



Using dendrograms to cluster observations - I

Dendrograms can be used to cluster observations into a discrete number of clusters or groups.

To do this we make a horizontal *cut* across the dendrogram, as shown in the top plots in the previous slide.

Here the cut point has been chosen to produce 3 clusters in each case.

The leaves of the dendrograms have been coloured to indicate cluster membership of the observations.

The clustering is also shown in 2D plots of the original observations, below each dendrogram.

Using dendrograms to cluster observations - II

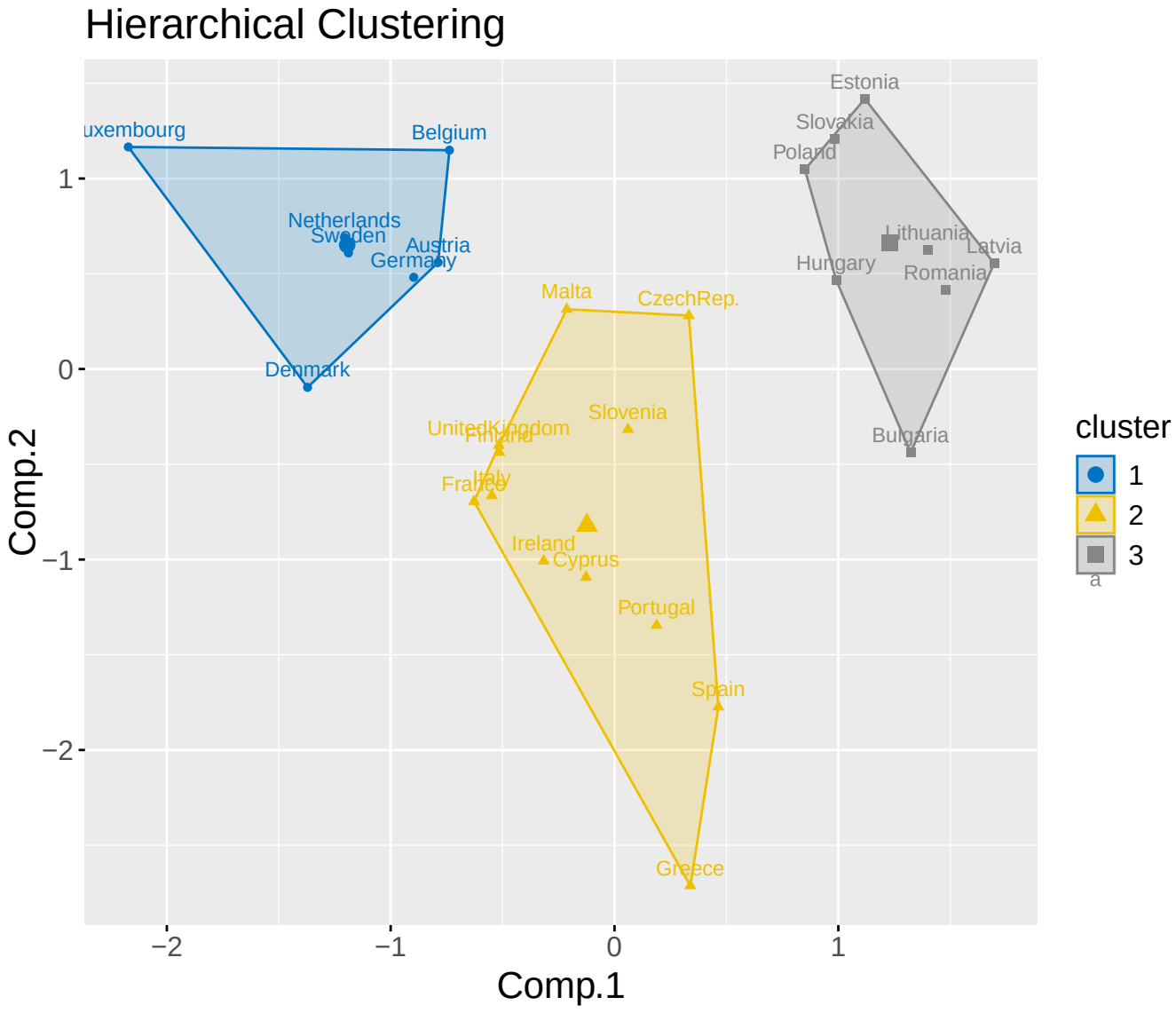
At this cut-point the Single Linkage method seems to have performed worse than Complete Linkage and Average Linkage, as it has a cluster that is just a single observation.

Complete Linkage and Average Linkage produce very similar clusterings, and differ only in their assignment of 1 observation.

It should be noted that the clustering produced by Single Linkage with 4 clusters produces very sensible results compared to the true clustering.

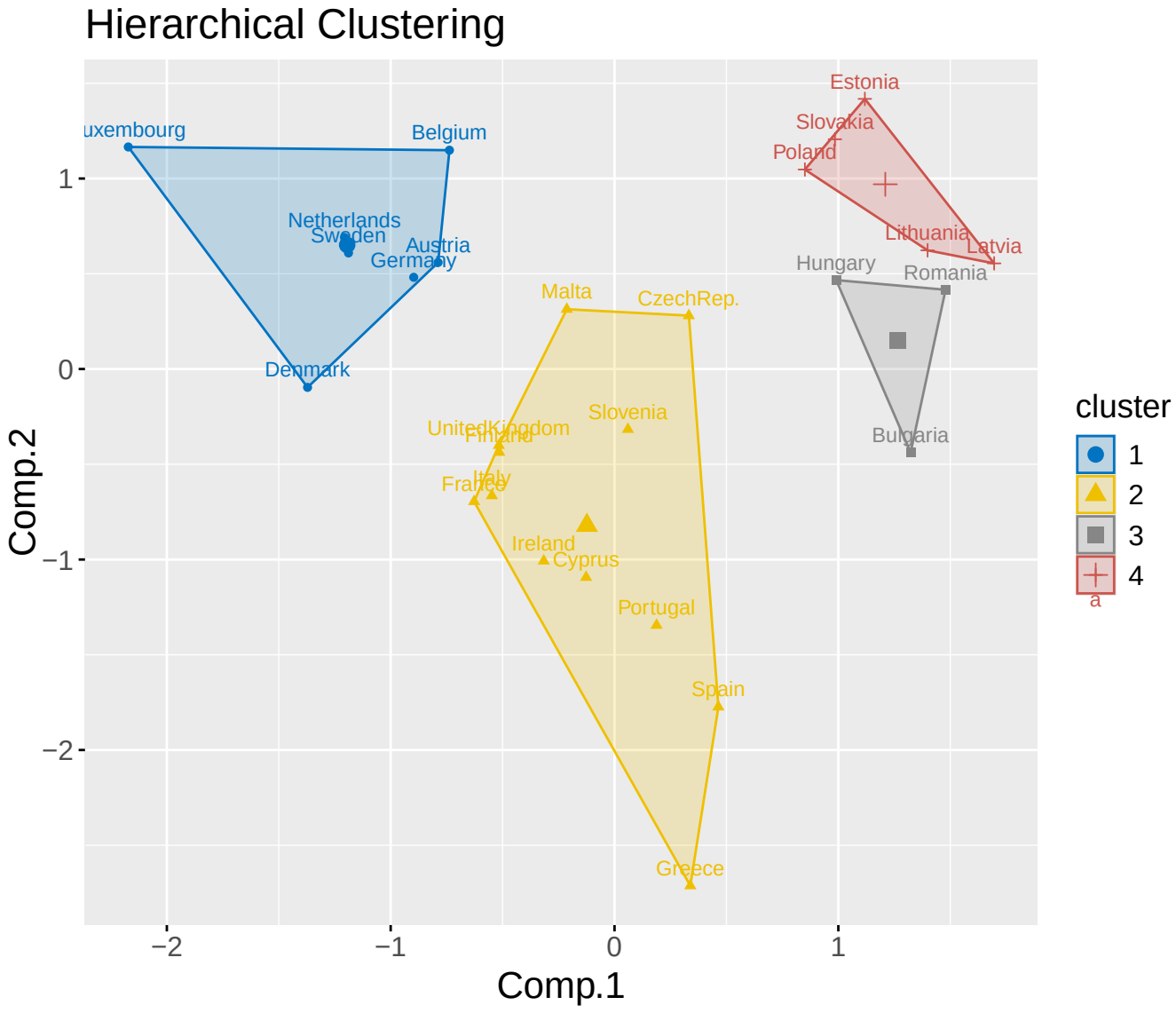
Hierarchical clustering - EU indicators

Hierarchical clustering - EU indicators



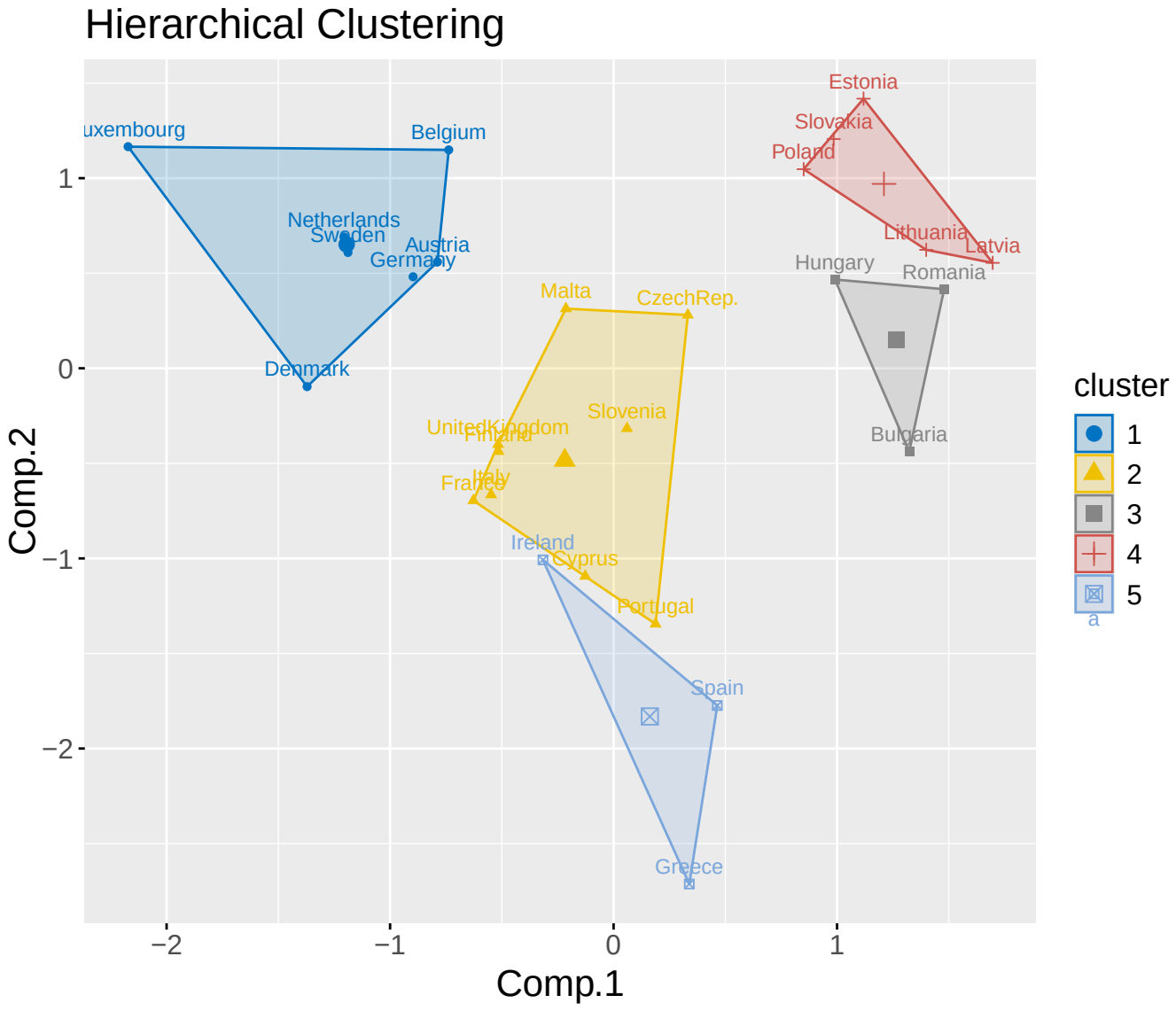
K-means clustering - EU indicators

Hierarchical clustering - EU indicators

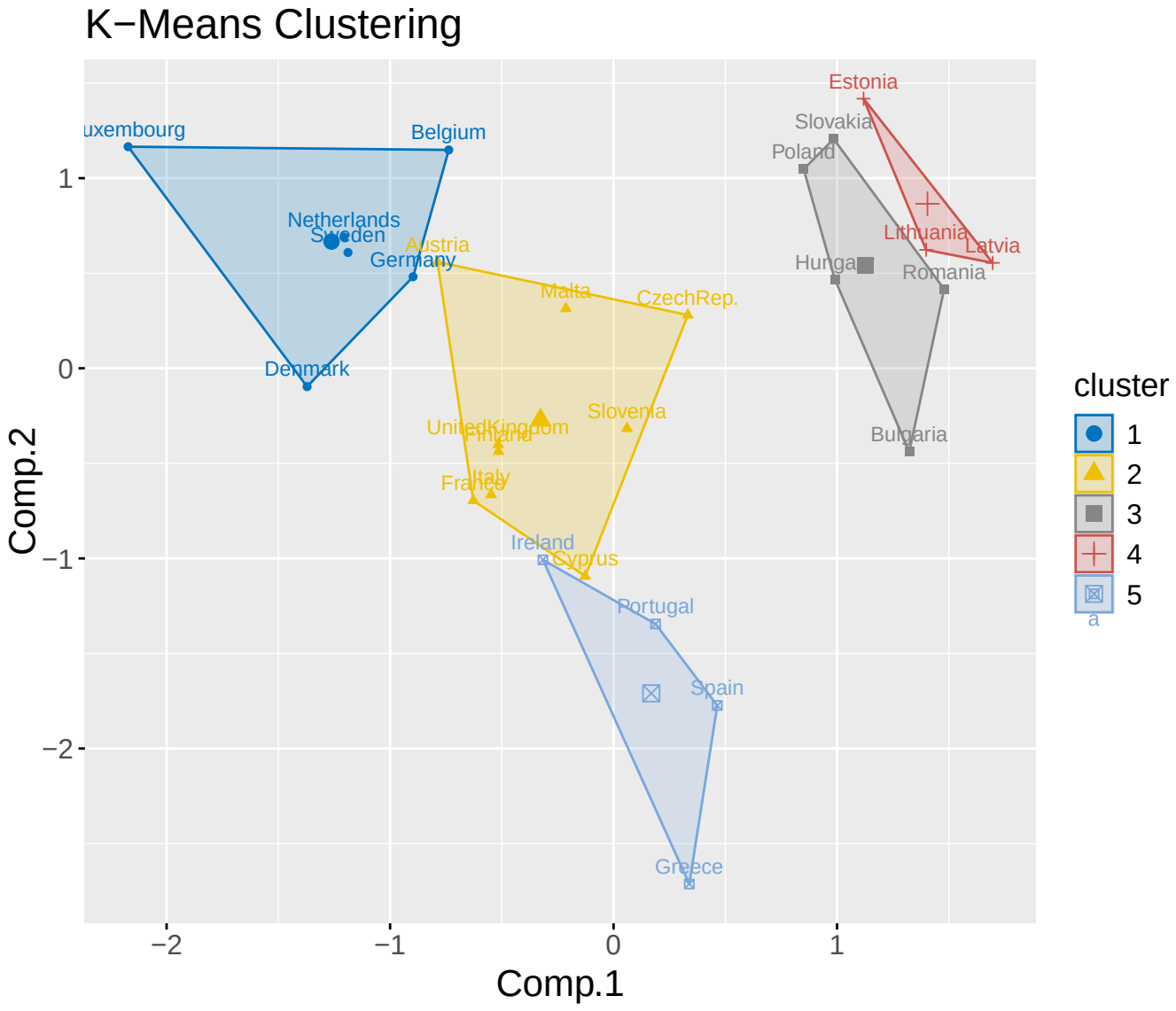


K-means clustering - EU indicators

Hierarchical clustering - EU indicators



Hierarchical clustering - EU indicators



Some papers with clustering appliecations: (clickable links)

The k-means Algorithm: A Comprehensive Survey and Performance Evaluation

K-means clustering using principal component analysis to automate label organization in multi-attribute seismic facies analysis

Biologics developability data analysis using hierarchical clustering accelerates candidate lead selection, optimization, and preformulation screening