

SIB
Swiss Institute of
Bioinformatics



Introduction to statistics

Isabelle Dupanloup, Rachel Marcone
February 2021

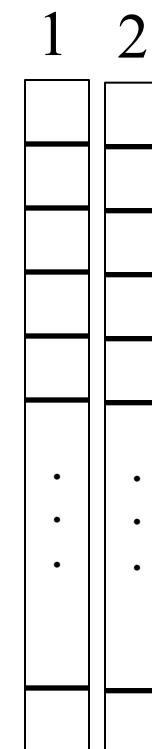
Visual representation

Dimension

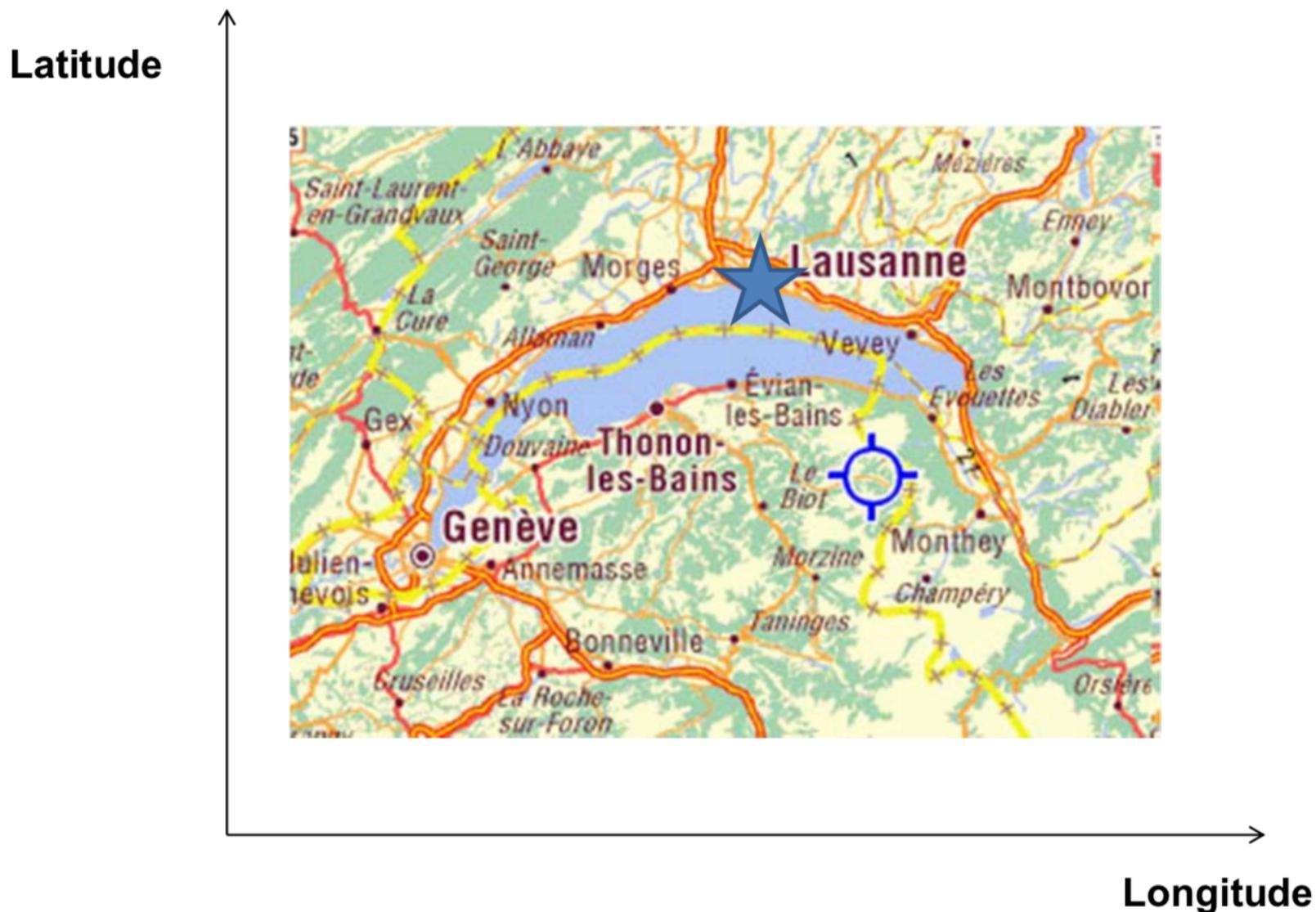
Dimension:
the number of coordinates we need to
locate a point in a given space.

2-dimension

	Coordinate1	Coordinate2
x1	1	2
x2	1	4
...
xN	5	2



Two dimensions: latitude and longitude

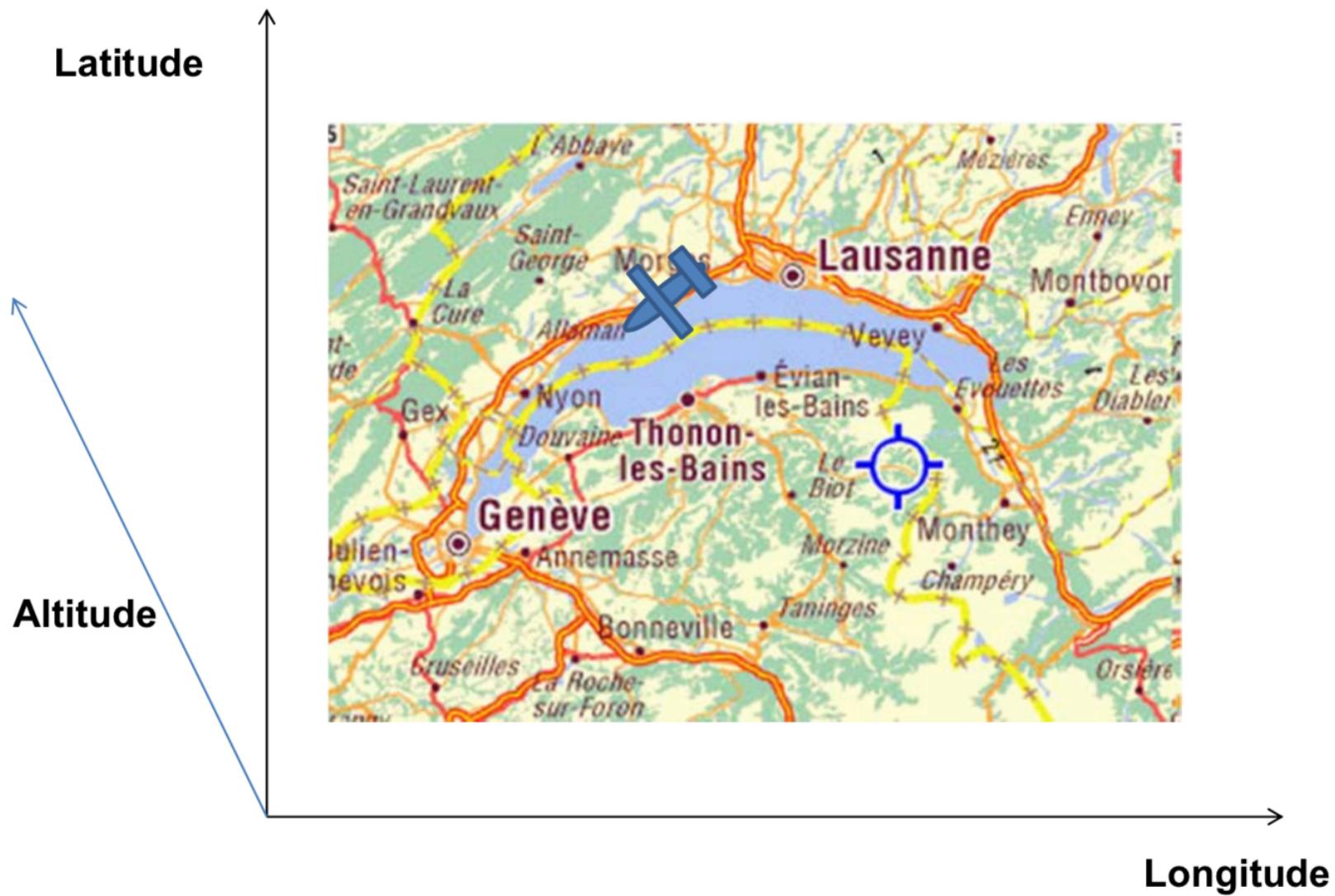


3-dimension

	Coordinate1	Coordinate2	Coordinate3
x1	1	2	5
x2	1	4	7
...
xN	5	2	1

1 2 3

Three dimensions: latitude, longitude and altitude



n-dimension

	Coordinate1	Coordinate2	Coordinate3	...	Coordinate-n	1	2	3	n
x1	1	2	5	...	18				
x2	1	4	7	...	2				
...
xN	5	2	1	...	4				

Dimension in biology?

Example: Peptide

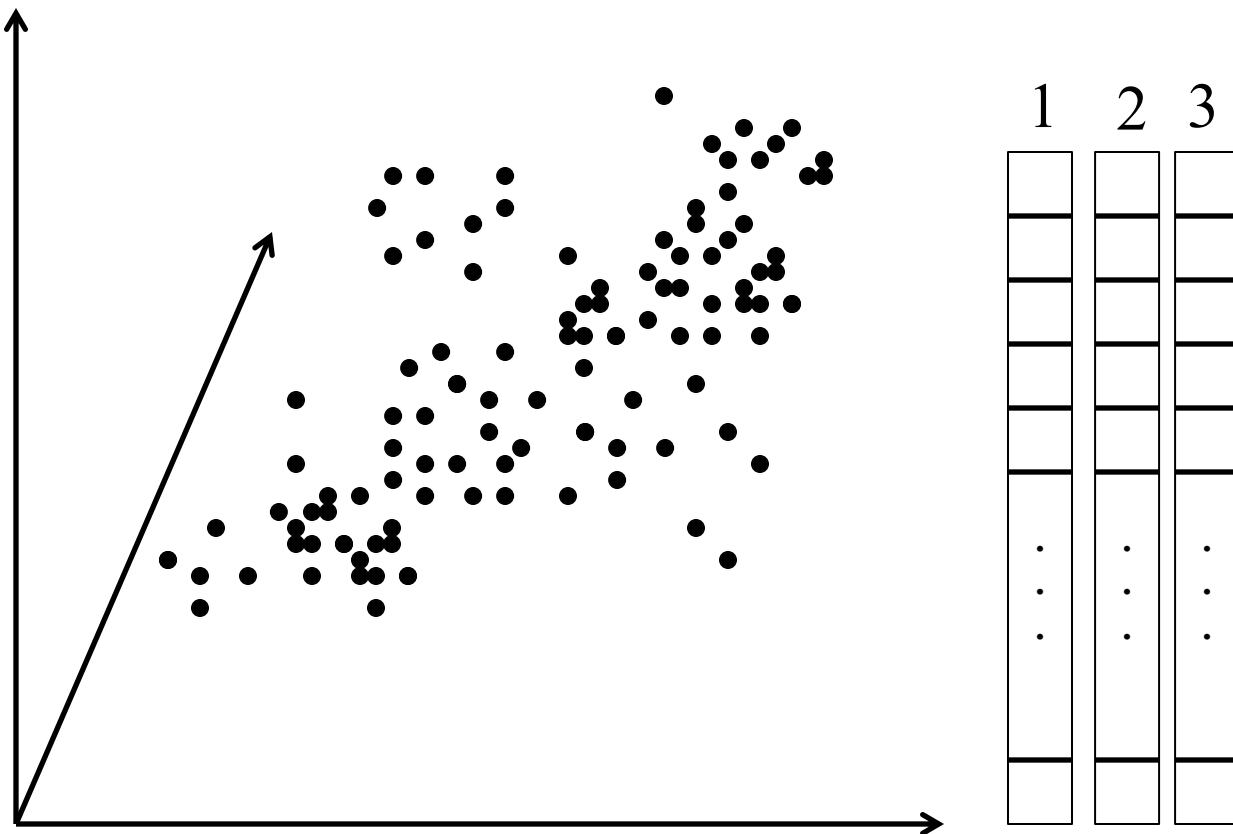
- peptide length
- peptide molecular weight
- peptide extinction coefficient
- peptide net charge at neutral pH
- peptide iso-electric point
- peptide water solubility

Dimension in biology?

Genes

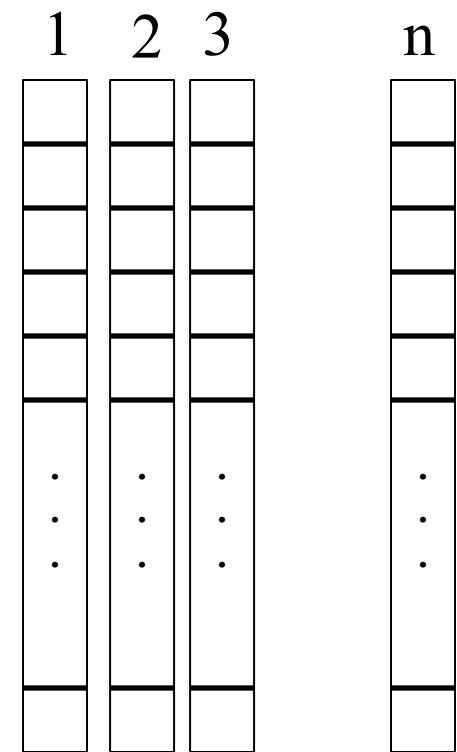
Dimension reduction

3-dimension

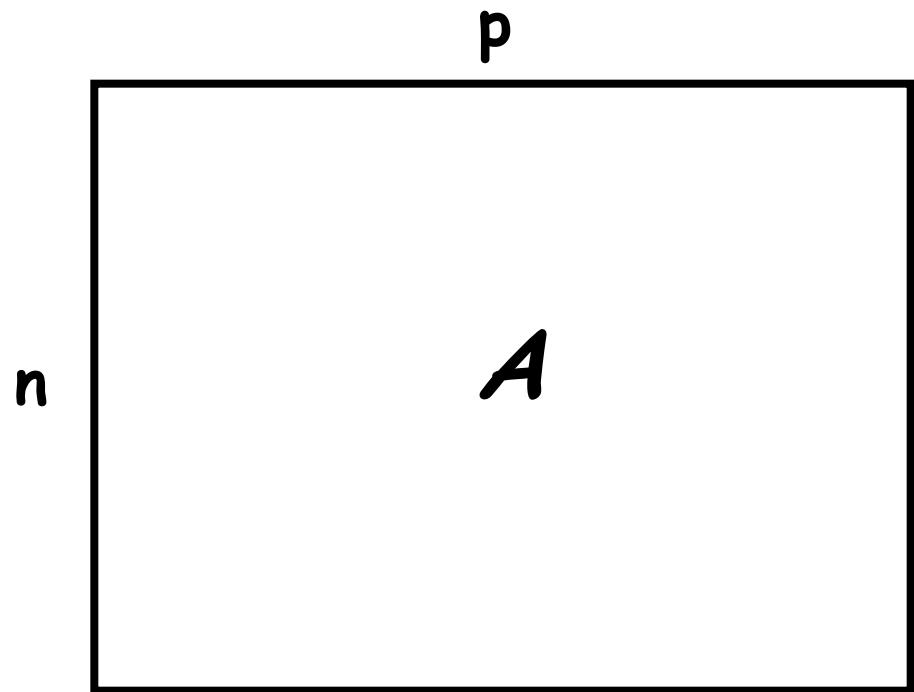


n-dimension

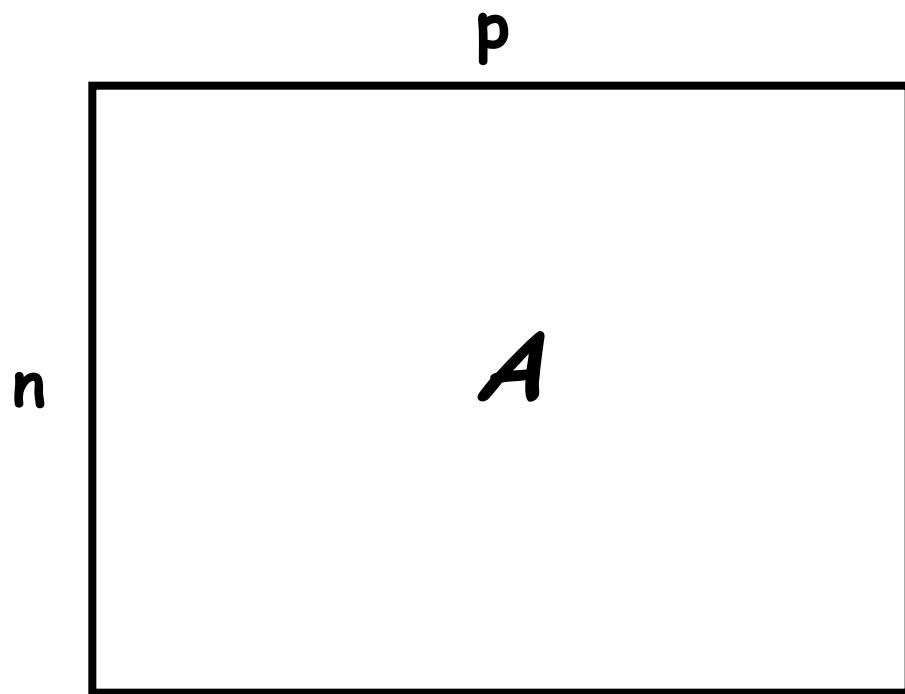
?



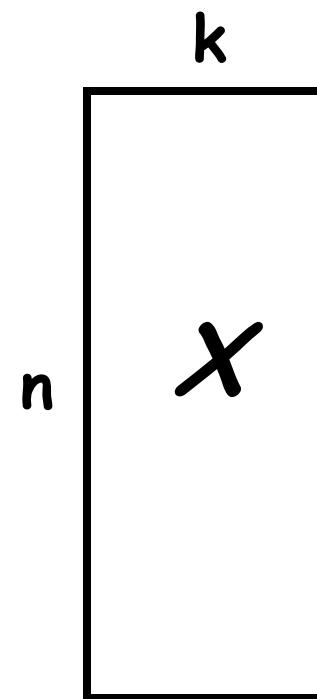
Starting point: Big Data

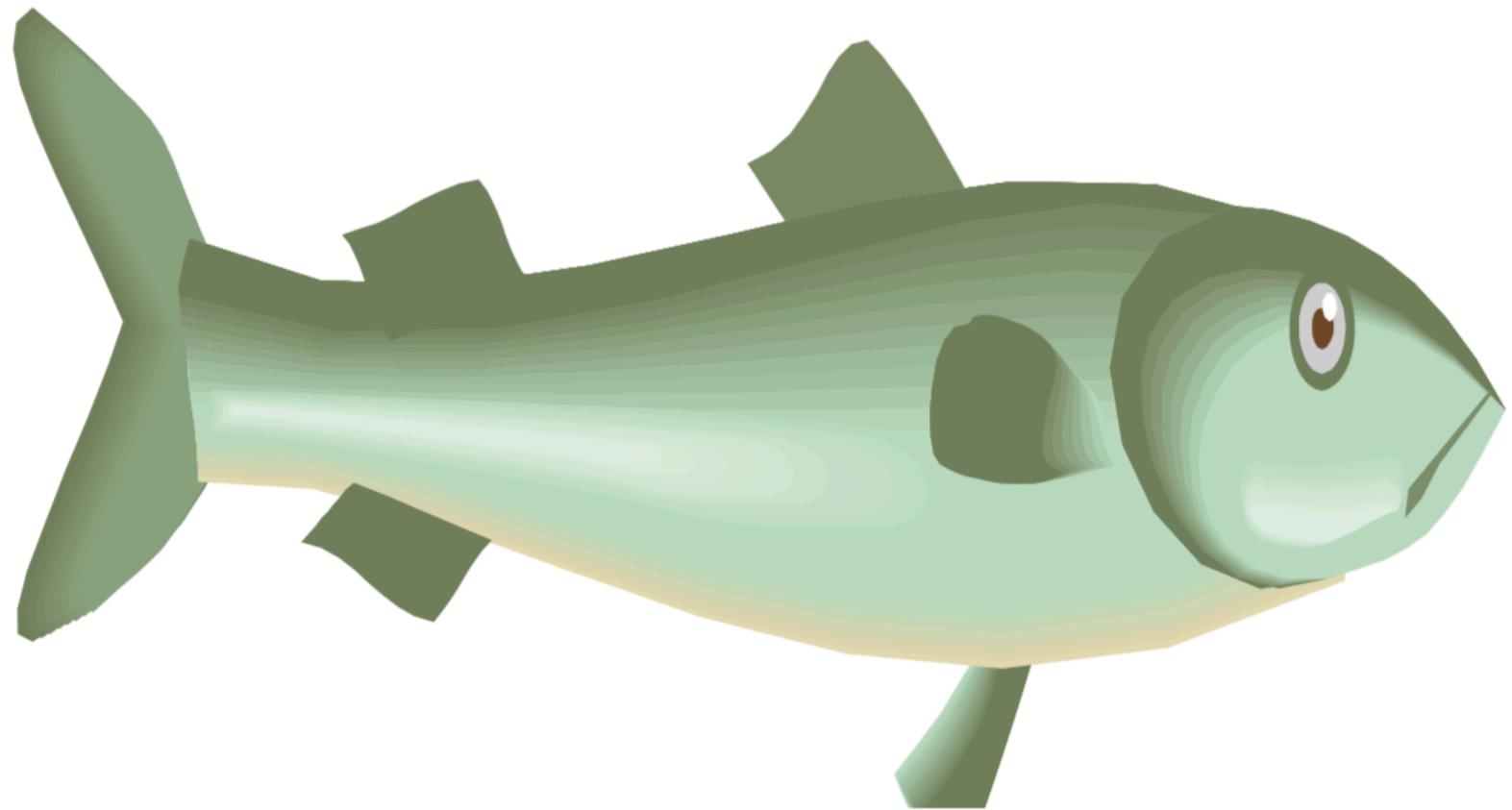


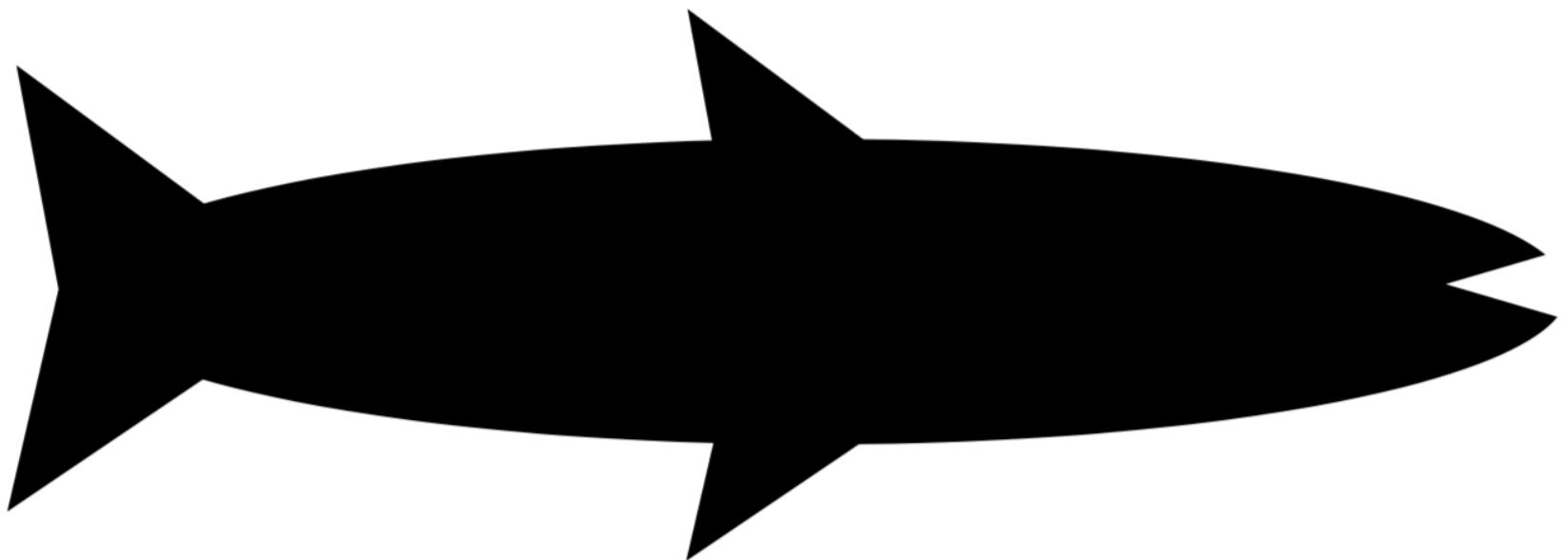
Starting point: Big Data



End result: human readable



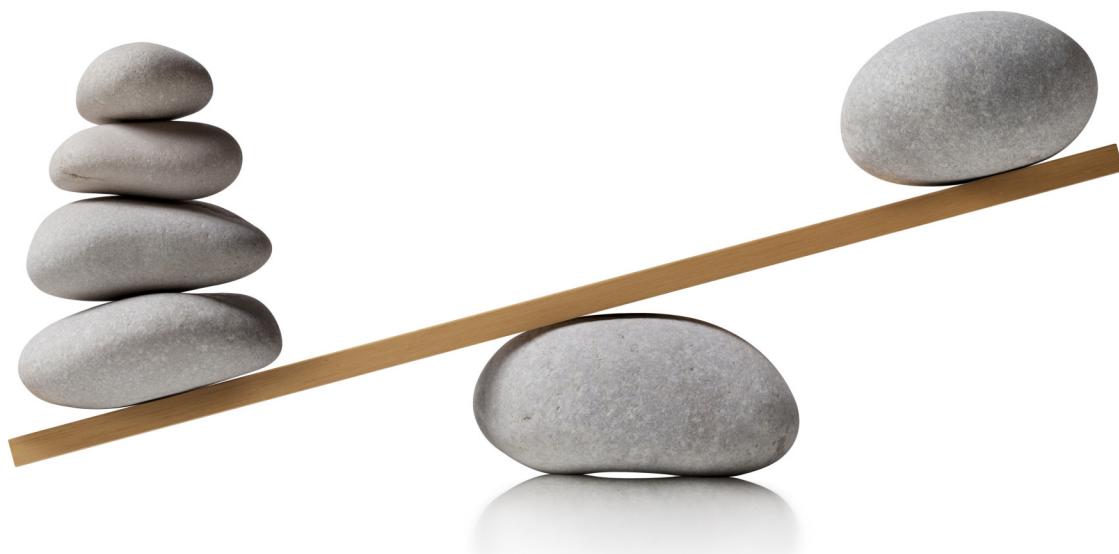






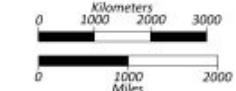
clarity of
representation

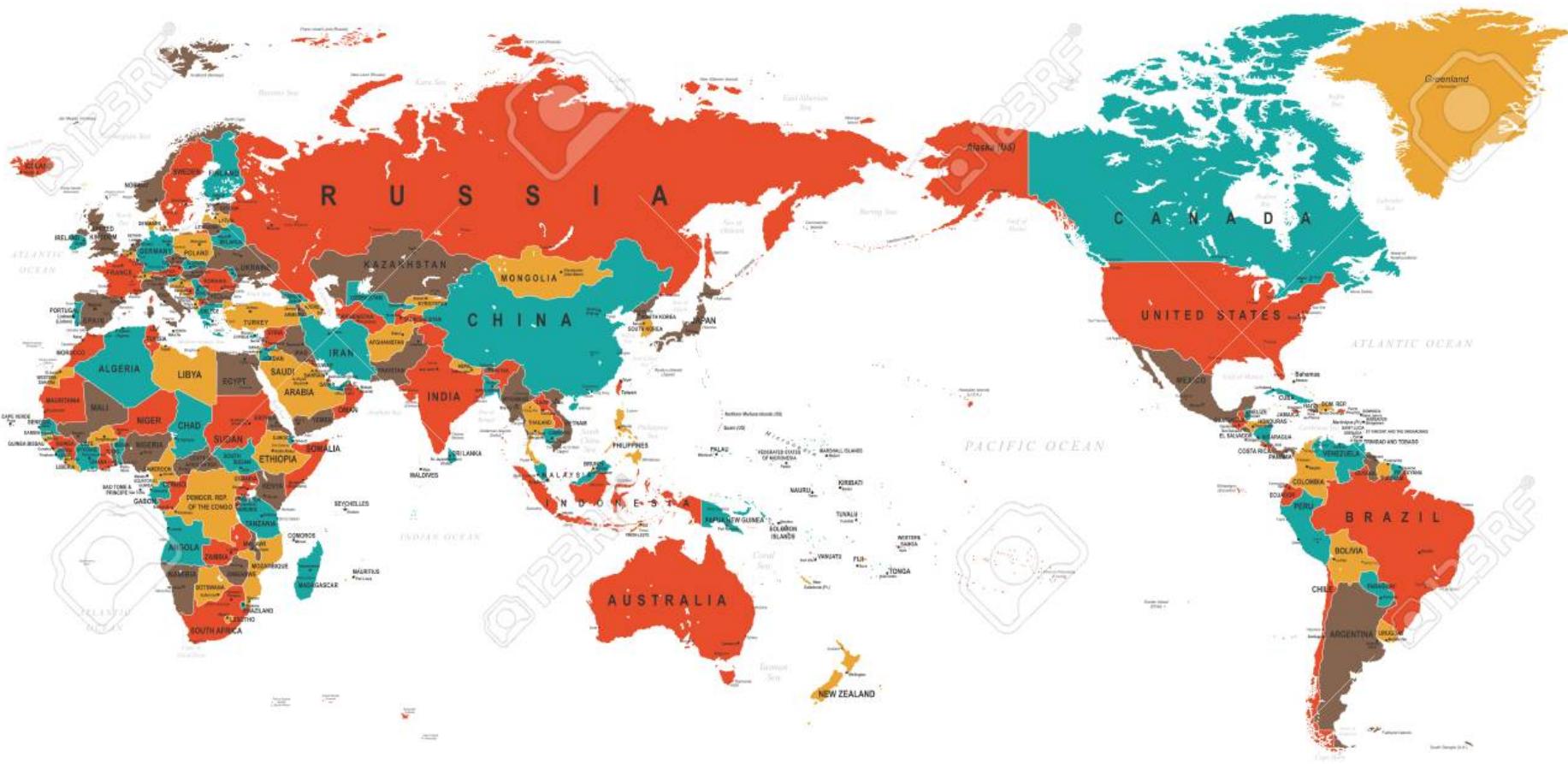
Over-simplification



**There are many possibilities and
there is not a « better » one than
another. It depends on what you want
to show.**

World Map



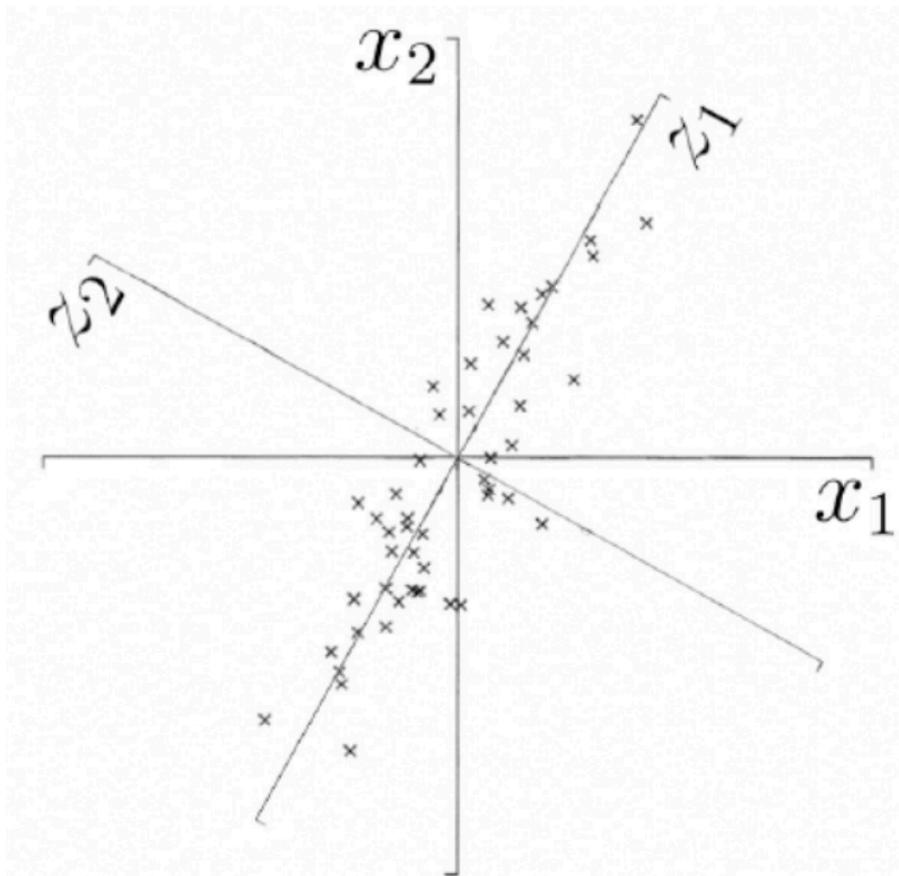
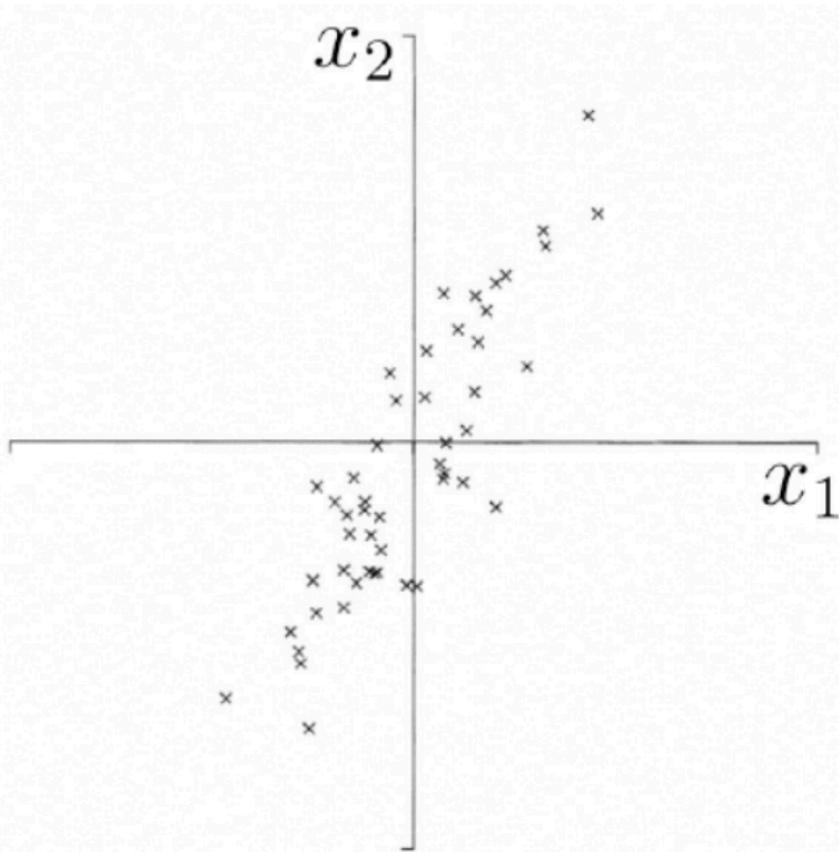


Principal Component Analysis (PCA)

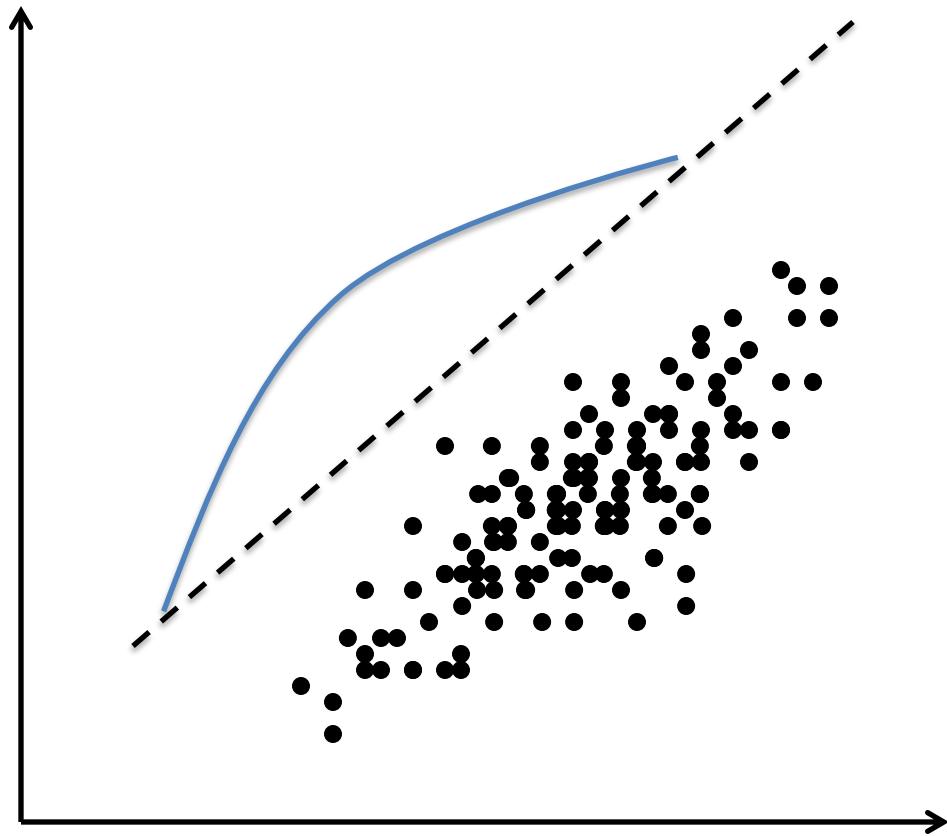
Pearson (1901) and Hotelling (1933)

- PCA is based on variance
- PCA is the best angle to see and evaluate the data

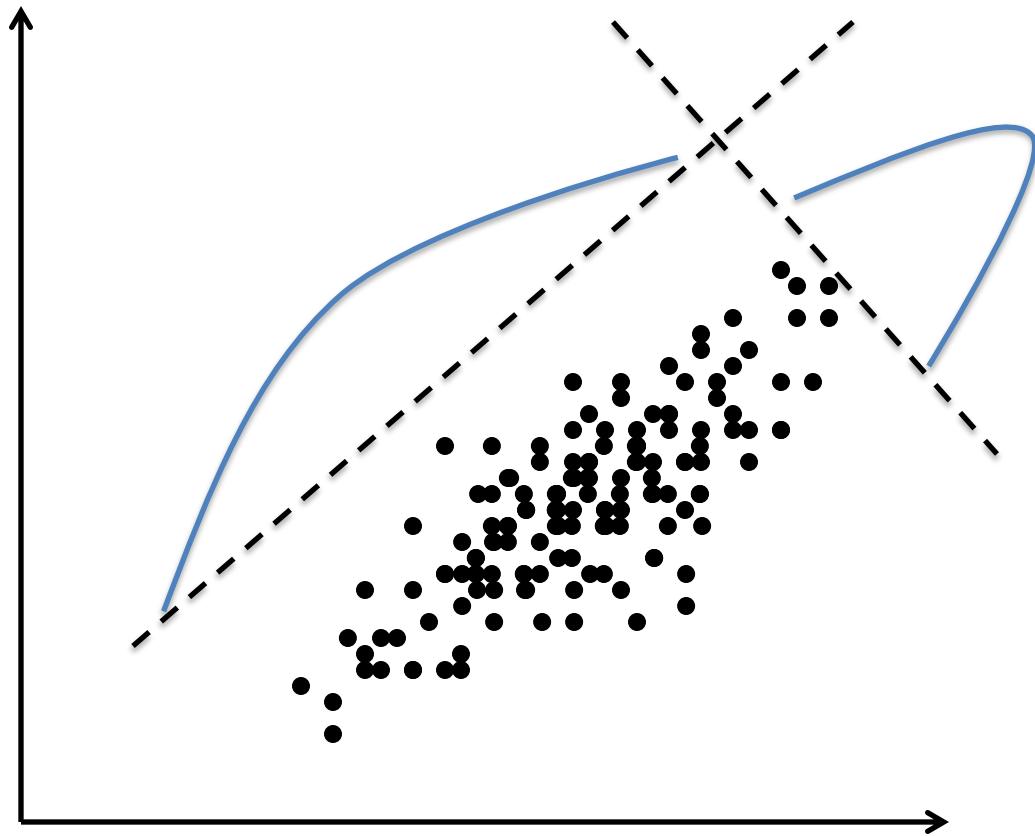
Which and how?



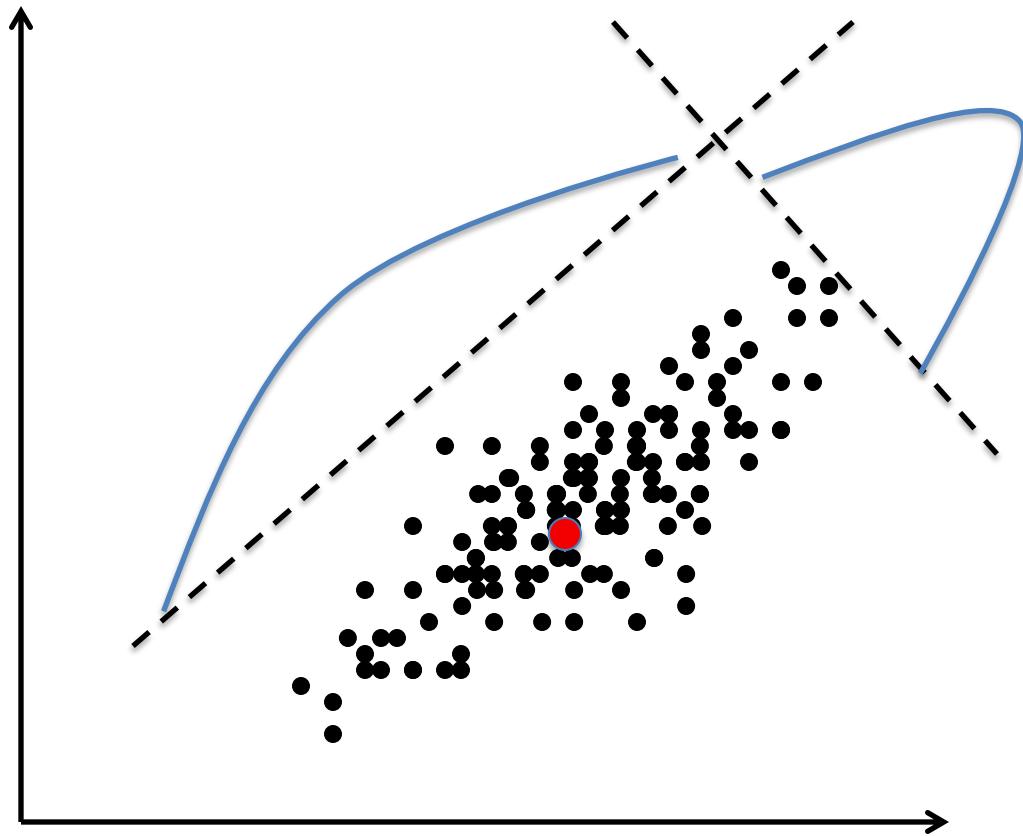
1. Largest variance first

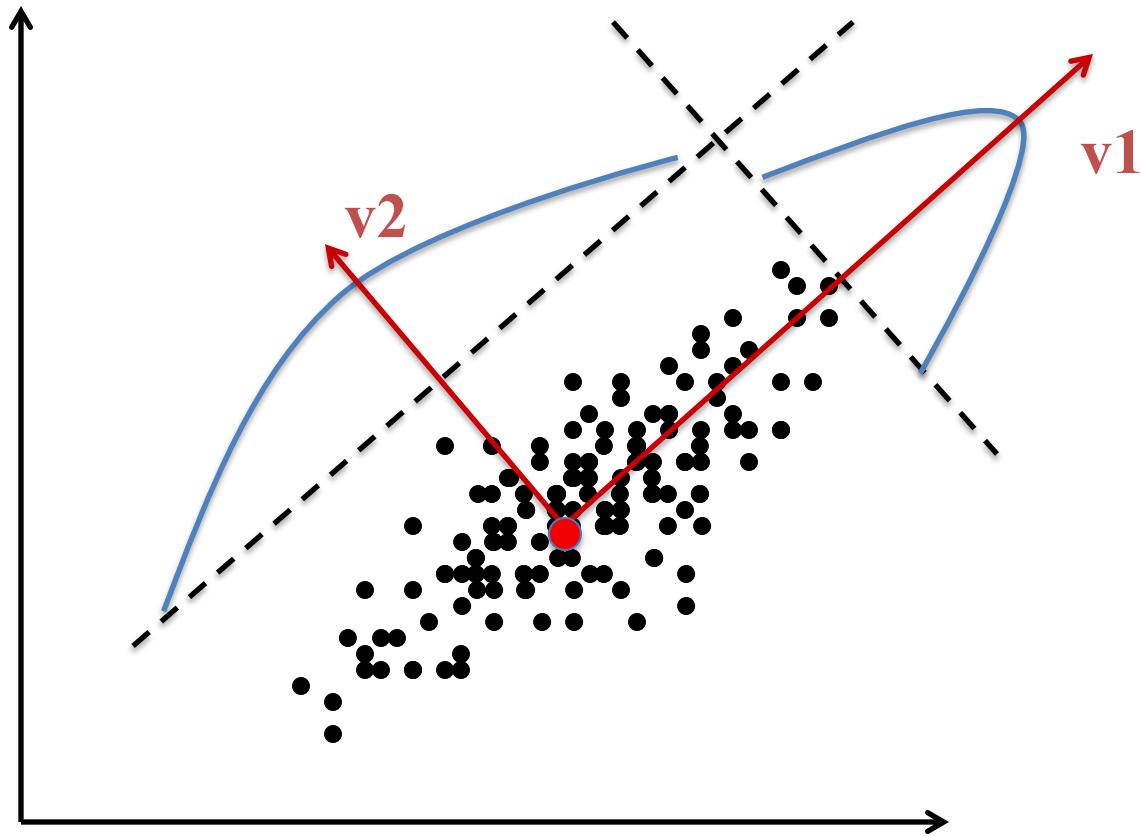


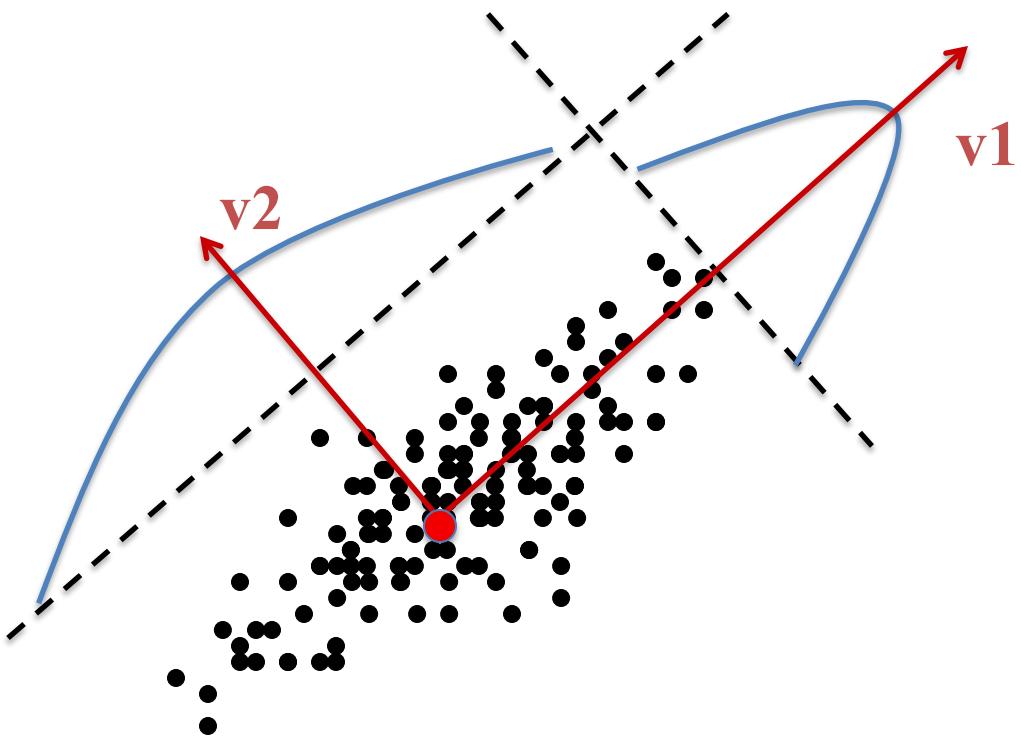
2. Select uncorrelated principal axis
(orthogonal)

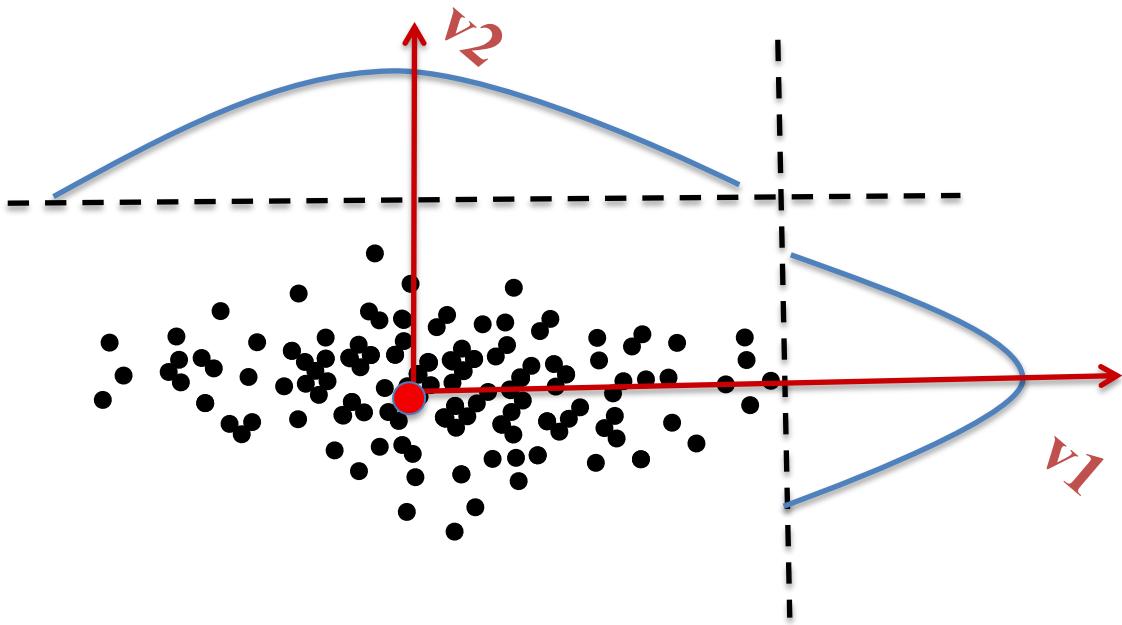


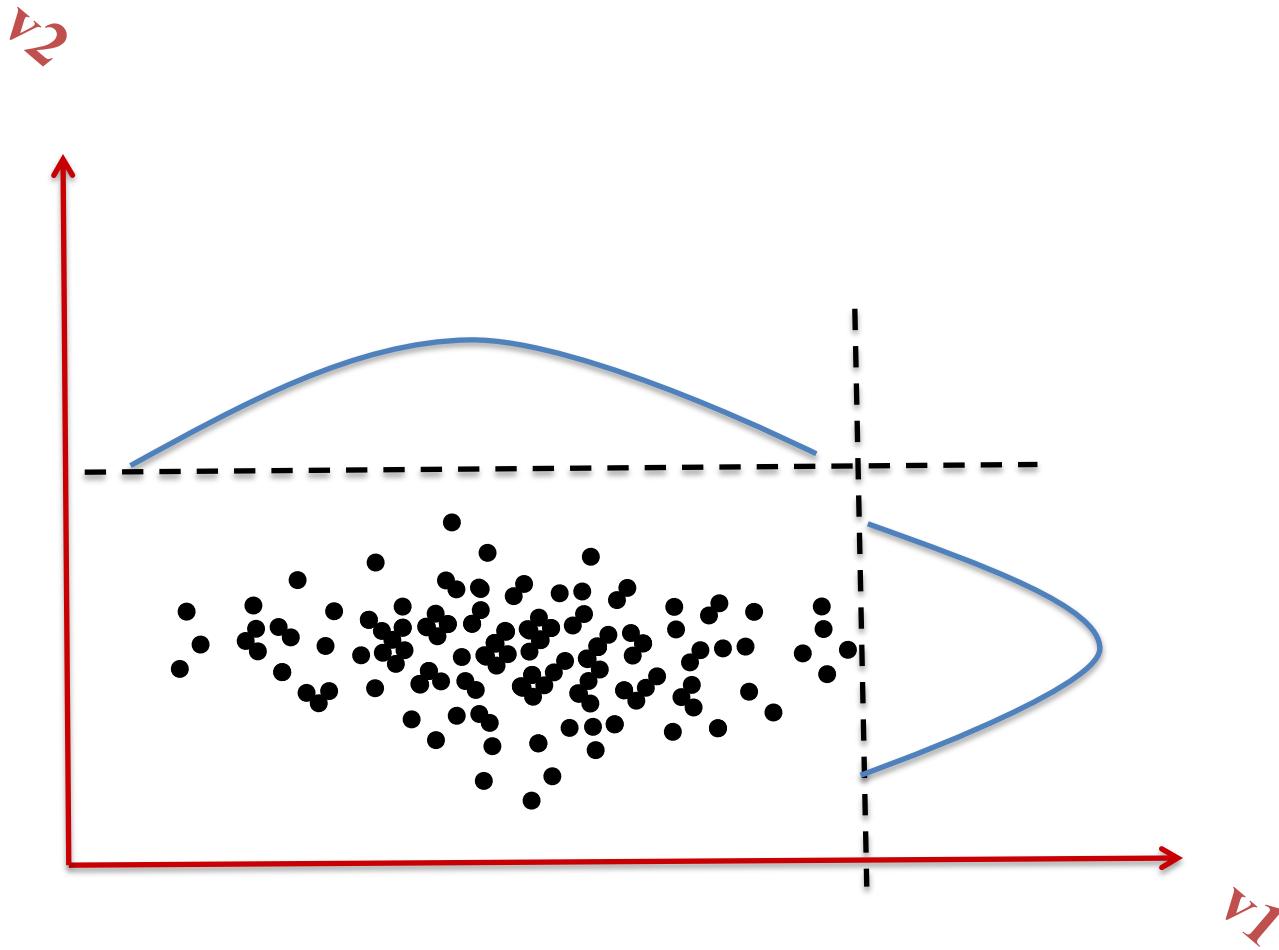
centroid



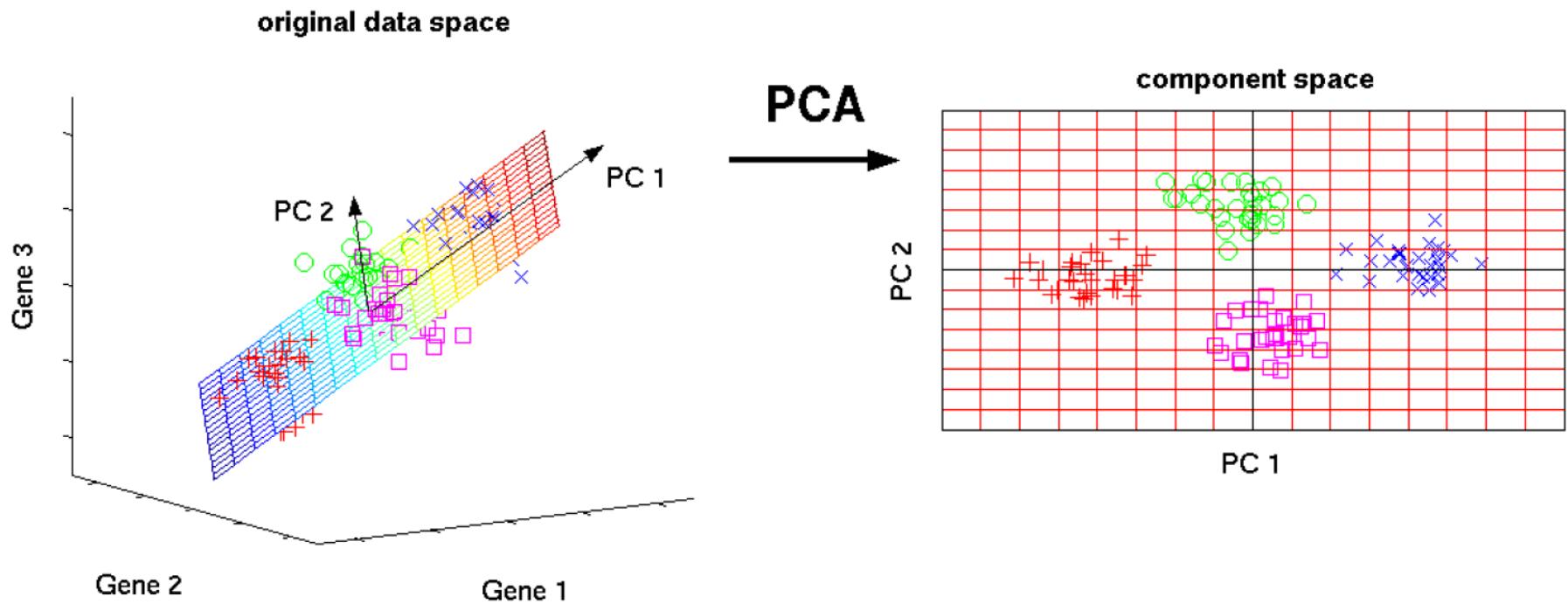








Without centroid

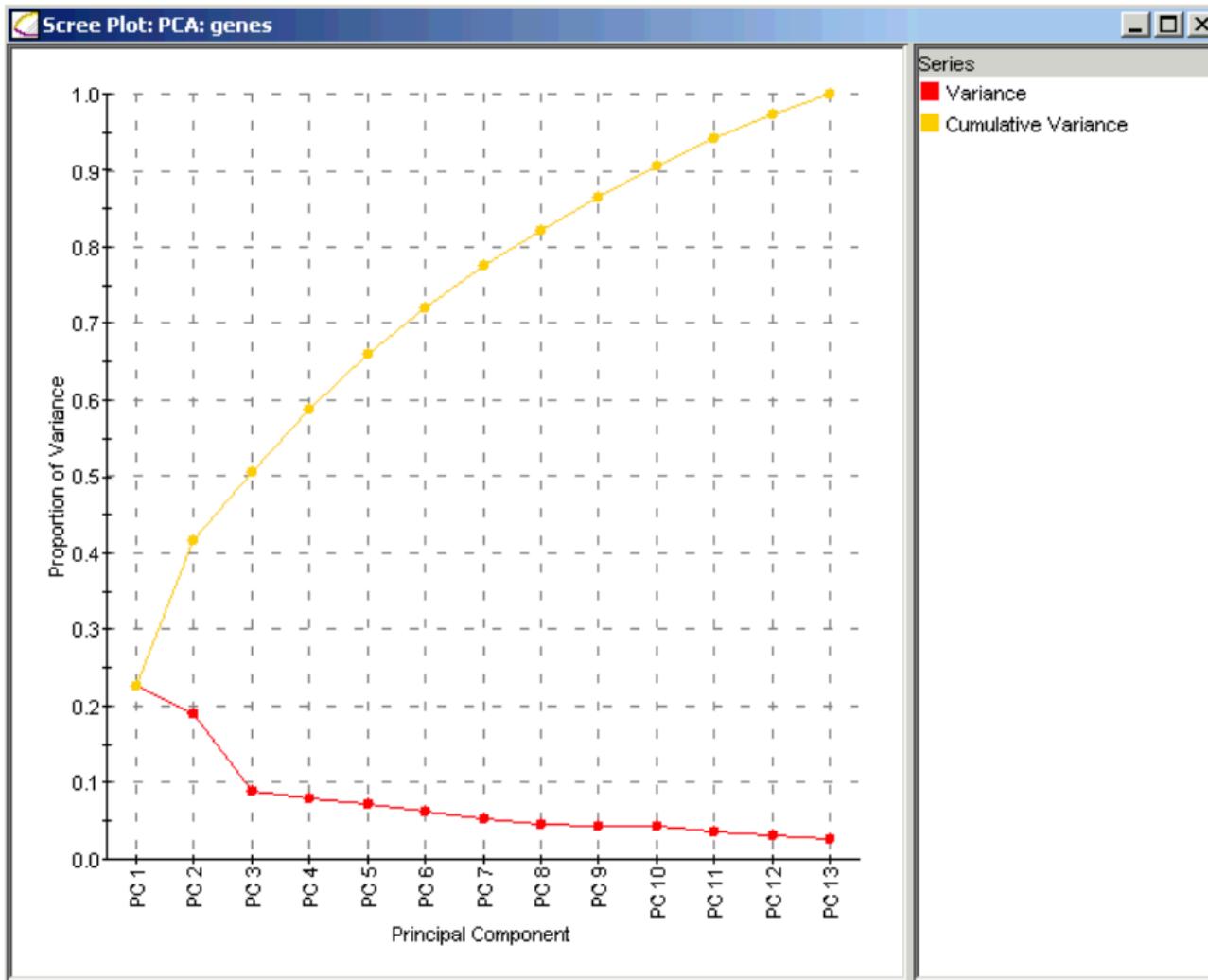


Mathematically

- Original coordinates (e_1, e_2, \dots, e_n)
- New coordinates are linear combinations of the original ones.
- You calculate the covariance matrix meaning a matrix containing two-by-two covariances
 - if positive then : the two variables increase or decrease together (correlated)
 - if negative then : One increases when the other decreases (Inversely correlated)
- $(\text{Corr}(X,Y) = \text{Cov}(X,Y)/\text{sd}(x)*\text{sd}(y))$
- Eigenvectors and eigenvalues are the linear algebra concepts that we need to compute from the covariance matrix in order to determine the ***principal components*** of the data.

Mathematically

- eigenvectors of the Covariance matrix are *the directions of the axes where there is the most variance (this is something you can prove mathematically!)*
- eigenvalues are the coefficients attached to eigenvectors, which give the *amount of variance carried in each Principal Component.*
- After having the principal components, to compute the percentage of variance (information) accounted for by each component, we divide the eigenvalue of each component by the sum of eigenvalues.



Scree Plot for Genetic Data. ([Source](#).)

<https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c>

How many PCs?

- Method 1: We **arbitrarily select** a number of principal components to include. Suppose I wanted to keep five principal components in my model. In the genetic data case above, these five principal components explains about 66% of the total variability that would be explained by including all 13 principal components.
- Method 2: Suppose I wanted to include **enough principal components to explain 90%** of the total variability explained by all 13 principal components. In the genetic data case above, I would include the first 10 principal components and drop the final three variables from Z^* .
- Method 3: Here, we want to “**find the elbow**.” In the scree plot above, we see there’s a big drop in proportion of variability explained between principal component 3 and the following. In this case, we’d likely include the first three features and drop the remaining features. As you can see, this method is a bit subjective as “elbow” doesn’t have a mathematically precise definition and, in this case, we’d include a model that explains only about 42% of the total variability.

In R

```
>pca<-prcomp(data, center = TRUE, scale. = FALSE)  
#coordinate of sample on components were identified
```

#Importance of components

```
>summary(pca)
```

In R

```
>pca<-prcomp(data, center = TRUE, scale. = FALSE)
#coordinate of sample on components were identified
```

#Importance of components

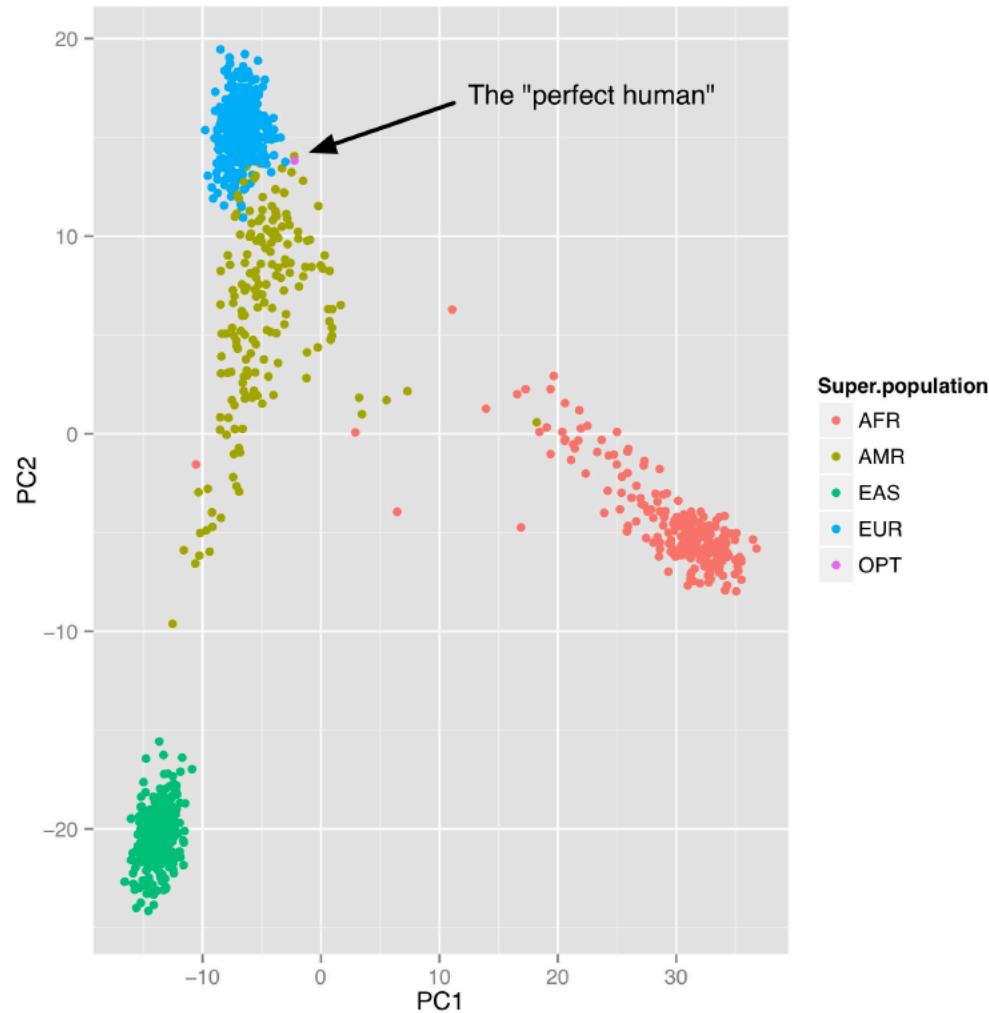
```
>summary(pca)
```

```
>pca$x
```

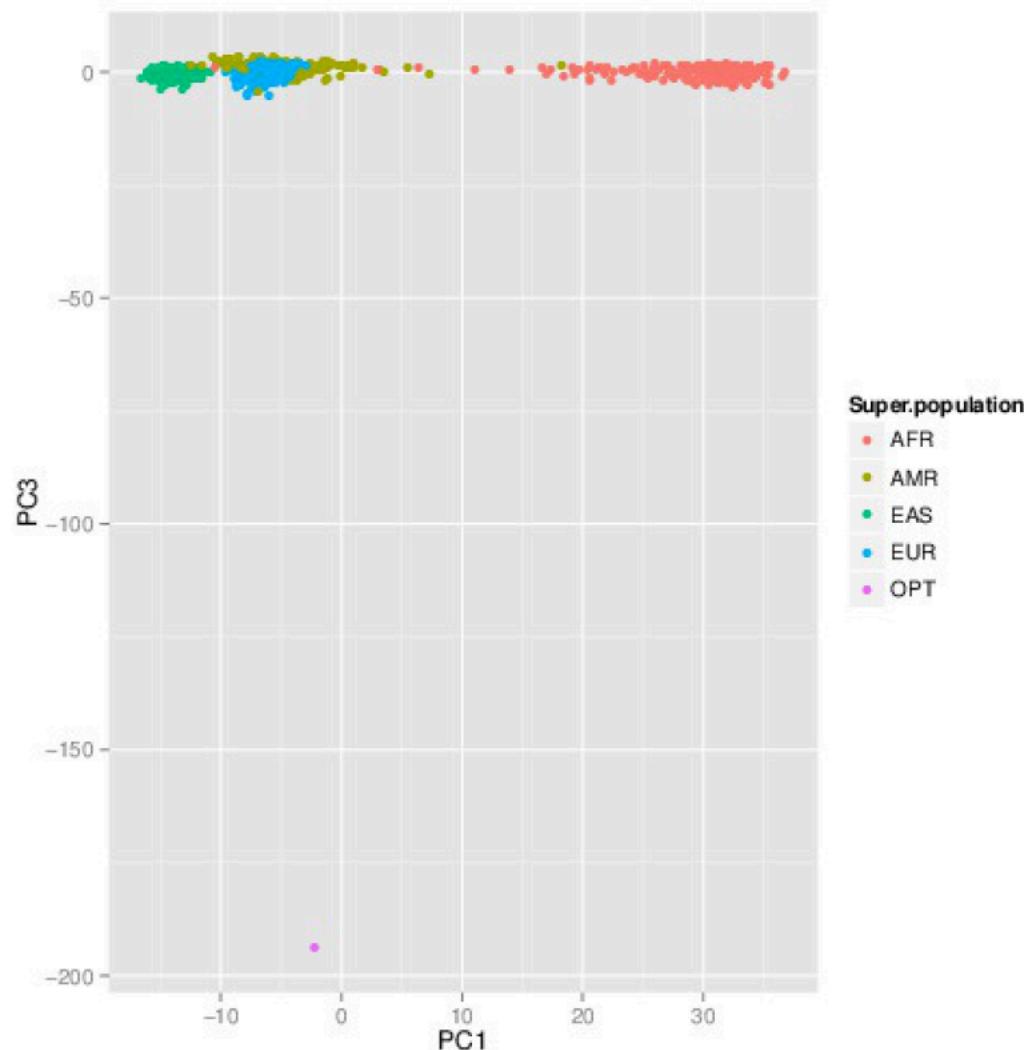
```
>plot(pca$x)
```

	PC1	PC2	PC3	PC4
[1,]	-2.684125626	-0.319397247	0.027914828	0.0022624371
[2,]	-2.714141687	0.177001225	0.210464272	0.0990265503
[3,]	-2.888990569	0.144949426	-0.017900256	0.0199683897
[4,]	-2.745342856	0.318298979	-0.031559374	-0.0755758166
[5,]	-2.728716537	-0.326754513	-0.090079241	-0.0612585926
[6,]	-2.280859633	-0.741330449	-0.168677658	-0.0242008576
[7,]	-2.820537751	0.089461385	-0.257892158	-0.0481431065
[8,]	-2.626144973	-0.163384960	0.021879318	-0.0452978706
[9,]	-2.886382732	0.578311754	-0.020759570	-0.0267447358
[10,]	-2.672755798	0.113774246	0.197632725	-0.0562954013
[11,]	-2.506947091	-0.645068899	0.075318009	-0.0150199245
[12,]	2.612255231	0.014720030	0.102150260	-0.1563702070

The perfect human is Puerto Rican...



... or an alien ?



What's new ?

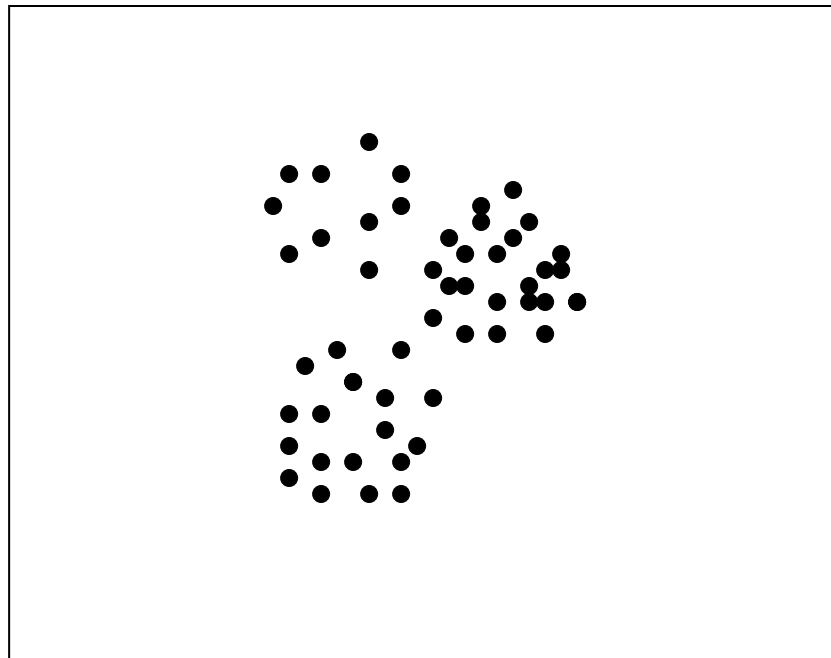
PCA	linear	Matrix Factorization		
ICA	linear	Matrix Factorization		
MDS	non-linear	Matrix Factorization		
Sparce NMF	non-linear	Matrix Factorization	2010	https://pdfs.semanticscholar.org/664d/40258f12ad28ed0b7d4c272935ad72a150db.pdf
cPCA	non-linear	Matrix Factorization	2018	https://doi.org/10.1038/s41467-018-04608-8
ZIFA	non-linear	Matrix Factorization	2015	https://doi.org/10.1186/s13059-015-0805-z
ZINB-WaVE	non-linear	Matrix Factorization	2018	https://doi.org/10.1038/s41467-017-02554-5

Diffusion maps	non-linear	graph-based	2005	https://doi.org/10.1073/pnas.0500334102
Isomap	non-linear	graph-based	2000	10.1126/science.290.5500.2319
t-SNE	non-linear	graph-based	2008	https://lvdmaaten.github.io/publications/papers/JMLR_2008.pdf
- BH t-SNE	non-linear	graph-based	2014	https://lvdmaaten.github.io/publications/papers/JMLR_2014.pdf
- Flt-SNE	non-linear	graph-based	2017	arXiv:1712.09005
LargeVis	non-linear	graph-based	2018	arXiv:1602.00370
UMAP	non-linear	graph-based	2018	arXiv:1802.03426
PHATE	non-linear	graph-based	2017	https://www.biorxiv.org/content/biorxiv/early/2018/06/28/120378.full.pdf

scvis	non-linear	Autoencoder (MF)	2018	https://doi.org/10.1038/s41467-018-04368-5
VASC	non-linear	Autoencoder (MF)	2018	https://doi.org/10.1016/j.gpb.2018.08.003

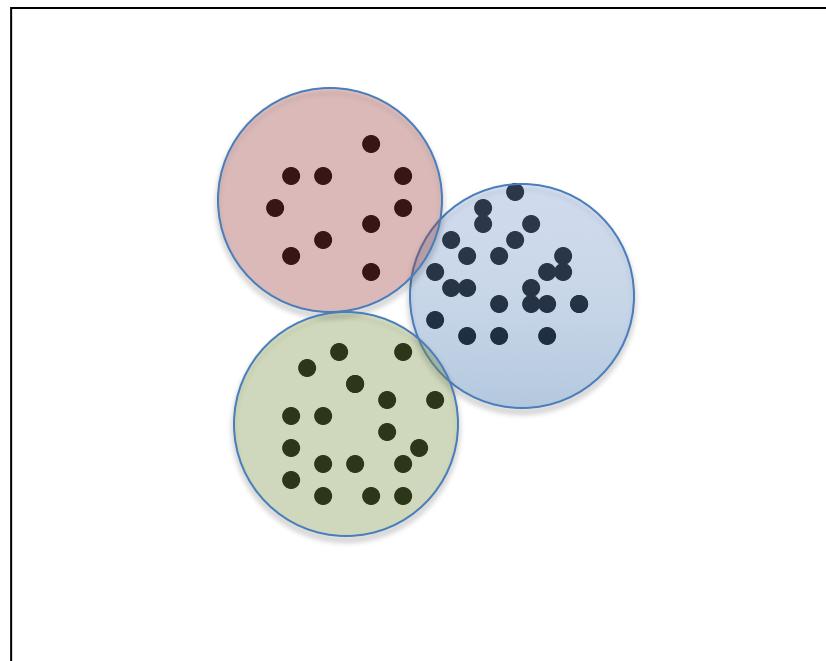
Clustering

Clustering



Point cloud

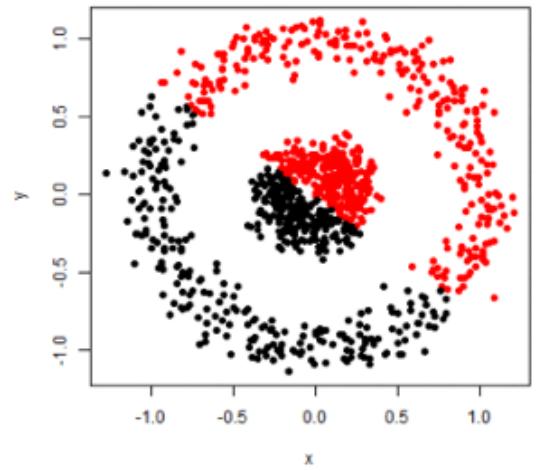
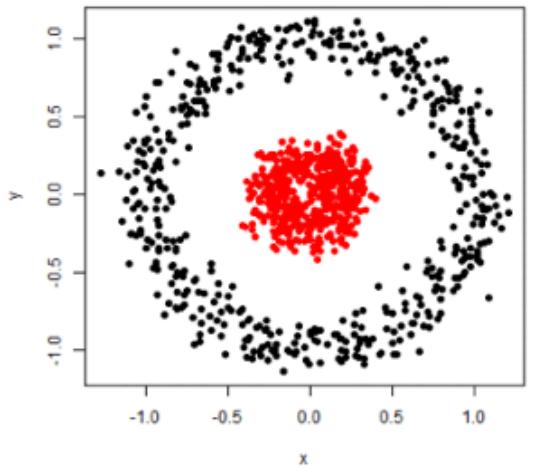
Clustering



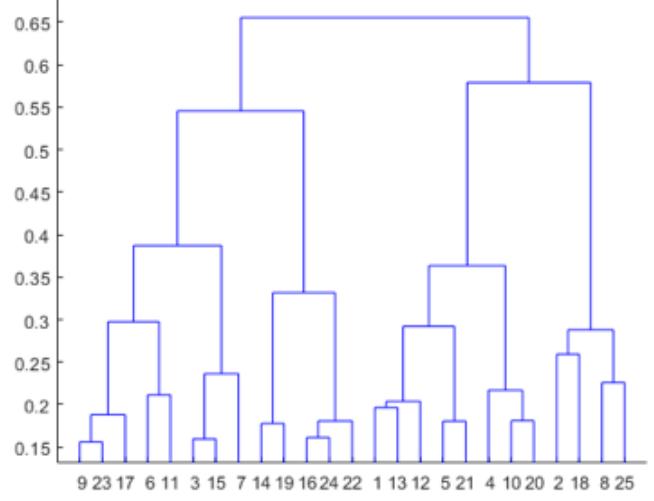
Clustering

Clustering method are divided into two categories :

Partitioning clustering



Hierarchical clustering



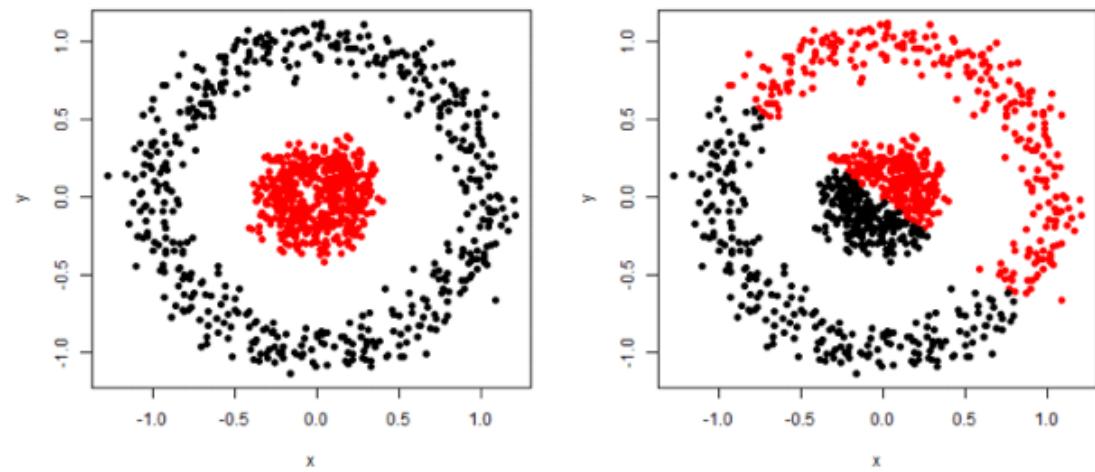
*Handbook of cluster analysis, Hennig C. et al.

Partitioning clustering

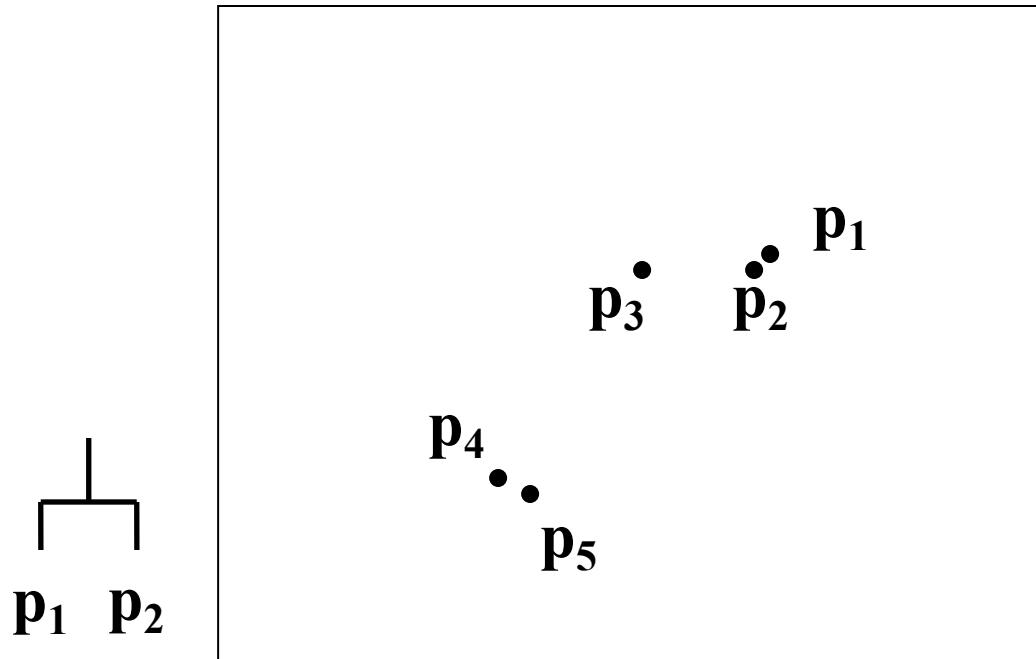
Convex partitioning. Example: K-means

Density based approaches. Example: DBSCAN

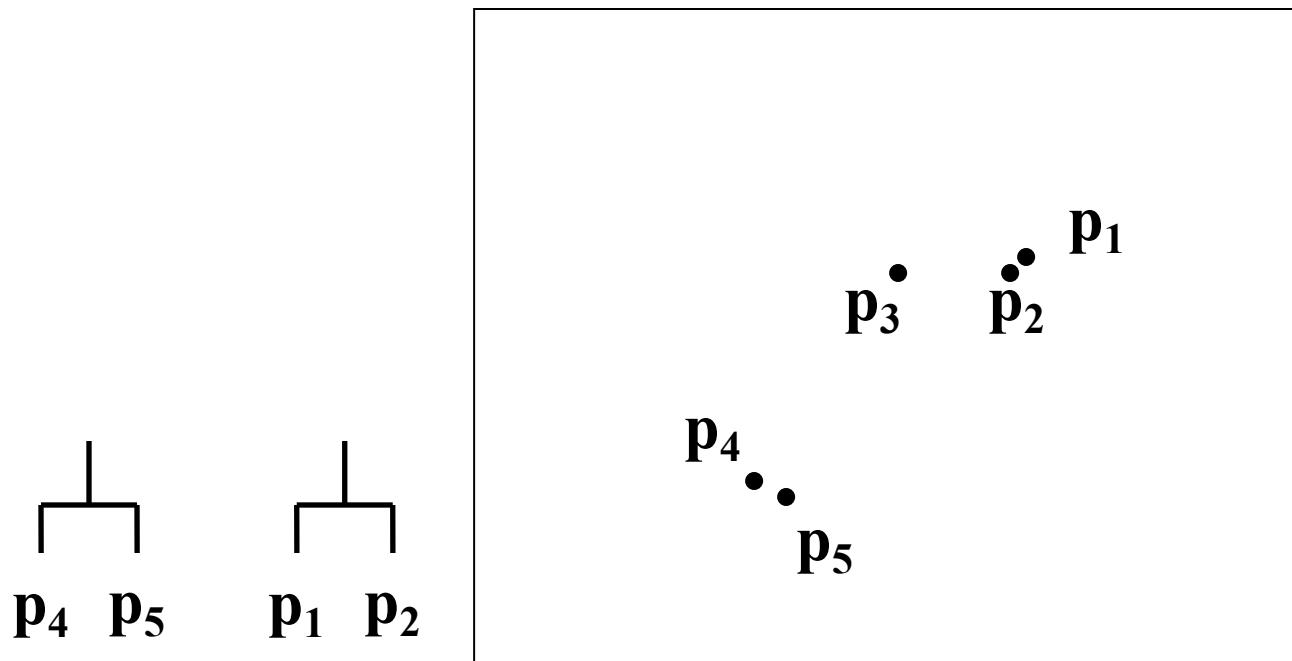
Model-based approaches. Example: Mclust



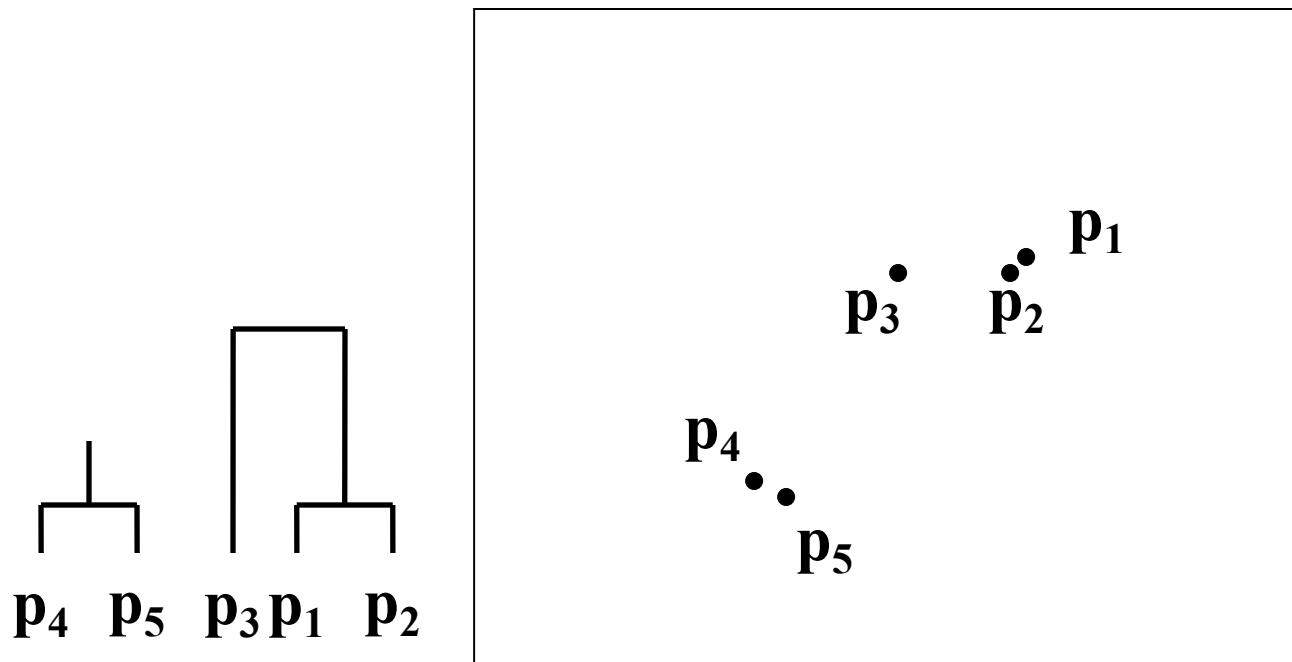
Hierarchical Clustering



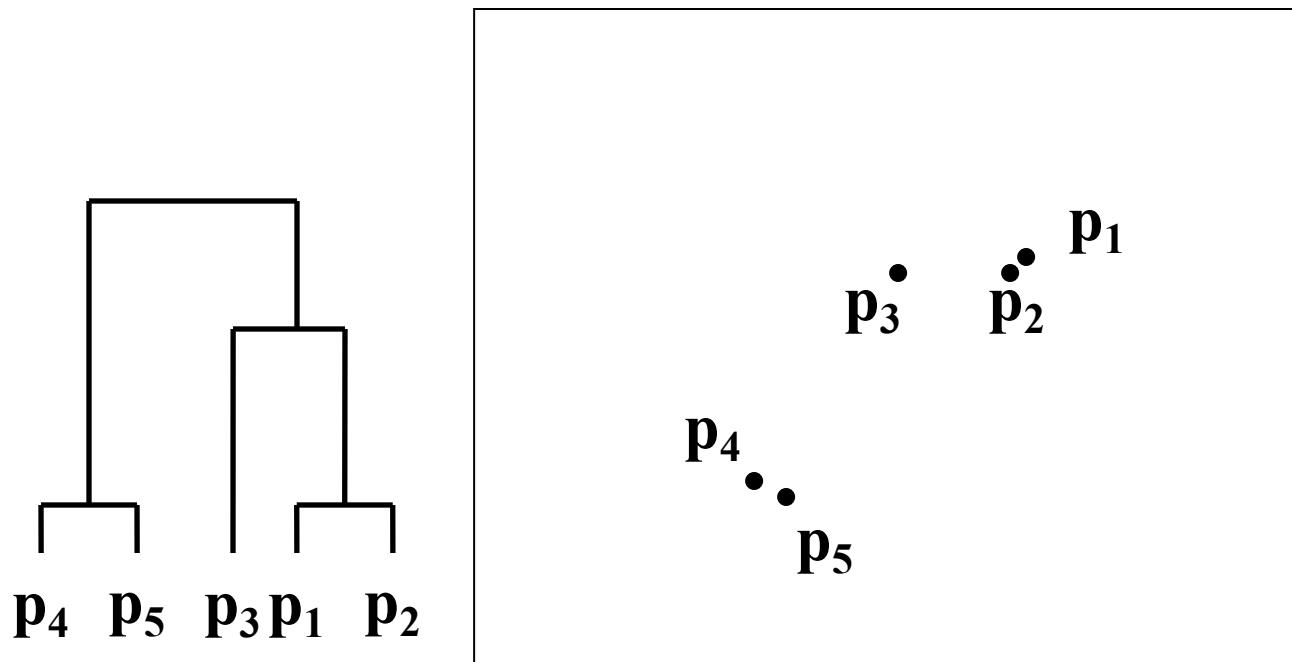
Hierarchical Clustering



Hierarchical Clustering



Hierarchical Clustering



Distance

Euclidean

$$X = (2, 0)$$

$$Y = (-2, -2)$$

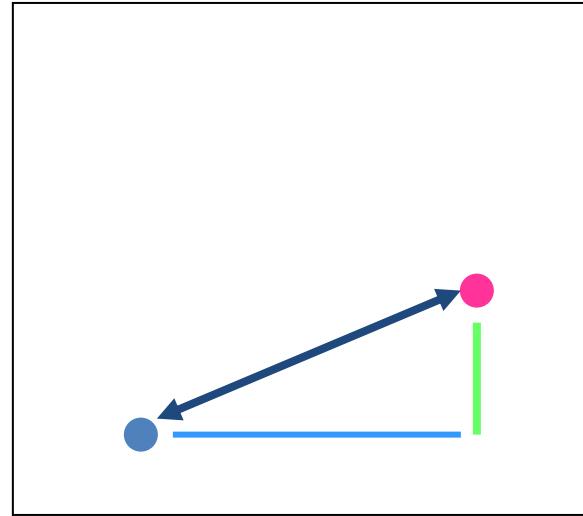
$$\sqrt{[\Sigma (y - x)^2]}$$

$$= \sqrt{(-2 - 2)^2 + (-2 - 0)^2}$$

$$= \sqrt{4^2 + 2^2}$$

$$= \sqrt{20}$$

$$= 4.47$$



It represents the “multivariate dissimilarity” of X & Y

Squared Euclidean

$$X = (2, 0)$$

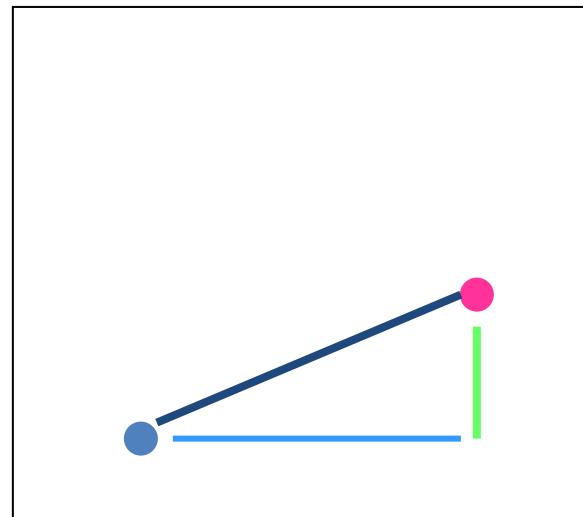
$$Y = (-2, -2)$$

$$\Sigma (y - x)^2$$

$$= (-2 - 2)^2 + (-2 - 0)^2)$$

$$= (4^2 + 2^2)$$

$$= 20$$



It represents the “multivariate dissimilarity” of X & Y

City Block

$$X = (2, 0)$$

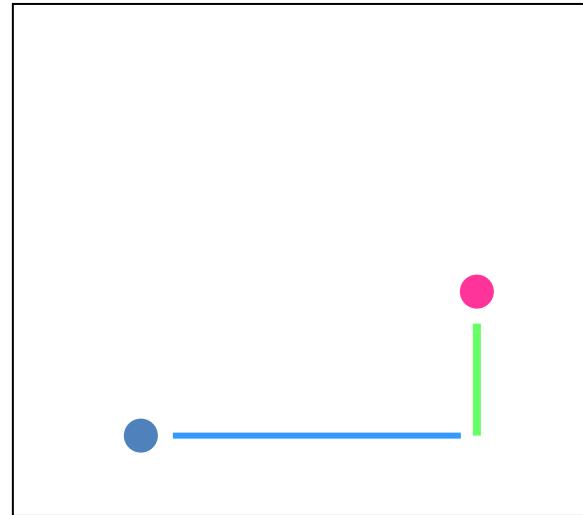
$$Y = (-2, -2)$$

$$\Sigma |y - x|$$

$$= (|-2 - 2| + |-2 - 0|)$$

$$= |-4| + |-2|$$

$$= 6$$



City Block-Manhattan distance

$$X = (2, 0)$$

$$Y = (-2, -2)$$

$$\sum |y - x|$$

$$= (|-2 - 2| + |-2 - 0|)$$

$$= | -4 | + | -2 |$$

$$= 6$$



Distance Measures in 2D

- Euclidean $\sqrt{[\sum (y - x)^2]}$
- Squared Euclidean $\sum (y - x)^2$
- City-Block $\sum |y - x|$

Distance Measures in nD

- Euclidean

$$d = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \dots + (a_n - b_n)^2}$$

- Squared Euclidean

$$d = (a_1 - b_1)^2 + (a_2 - b_2)^2 + \dots + (a_n - b_n)^2$$

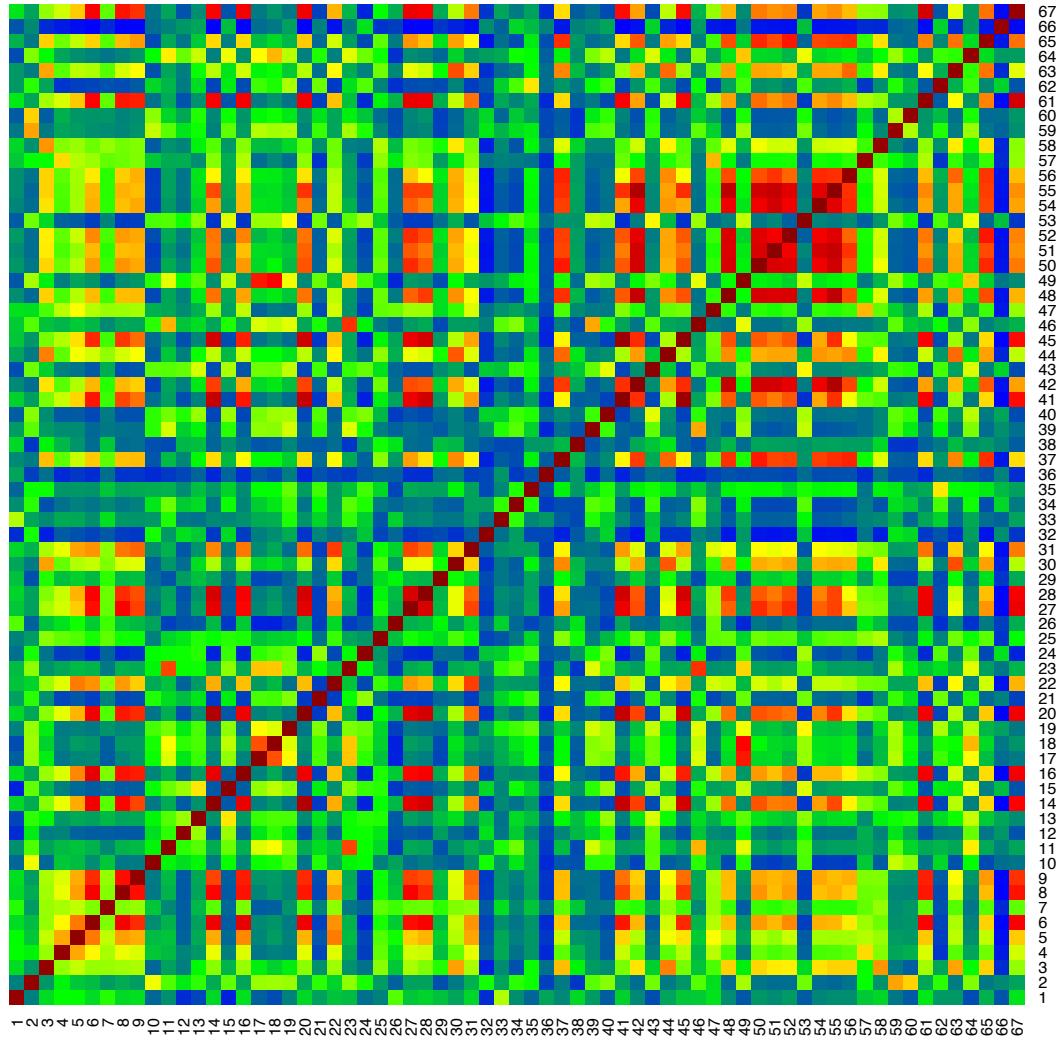
- City-Block

$$d = |a_1 - b_1| + |a_2 - b_2| + \dots + |a_n - b_n|$$

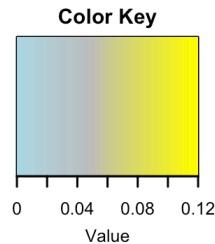
In R

>?dist

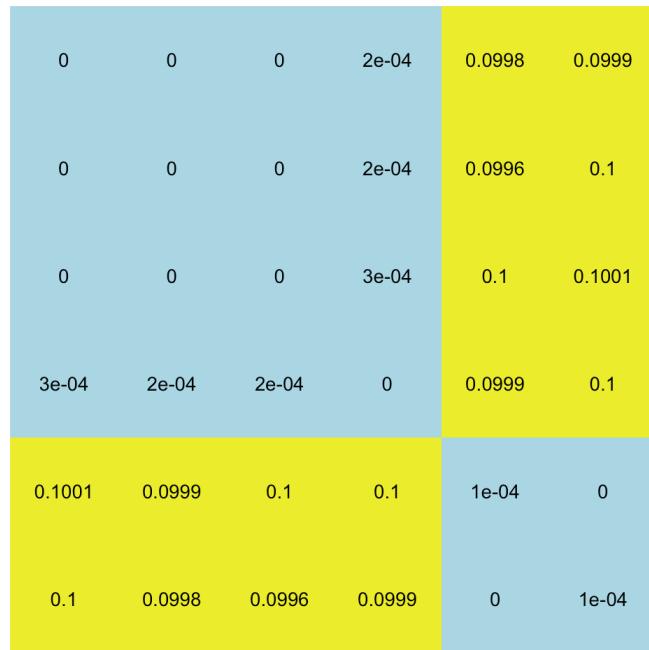
Distance matrix



Distance matrix



Heatmap of distance matrix



Sample 1

Sample 2

Sample 3

Sample 4

Sample 5

Sample 6

Sample 1

Sample 2

Sample 3

Sample 4

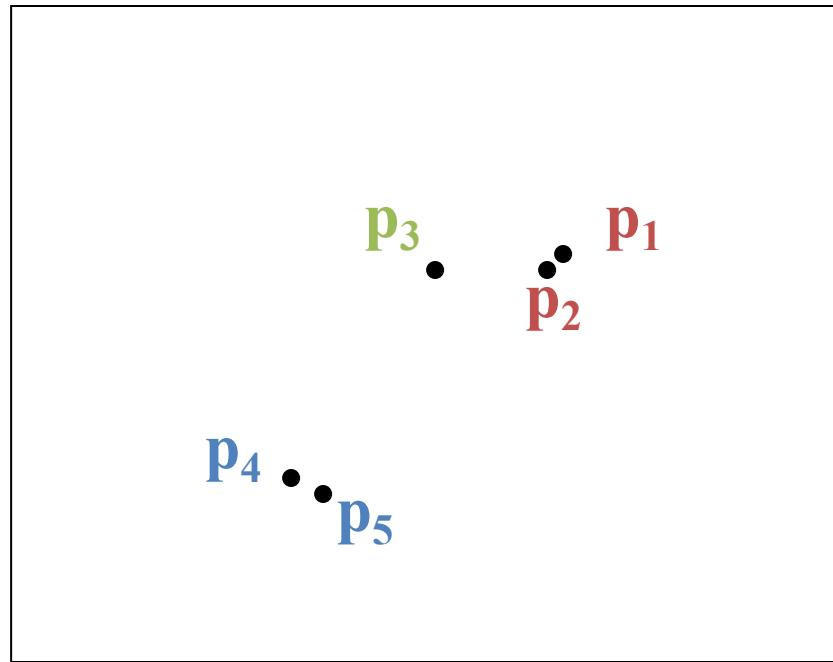
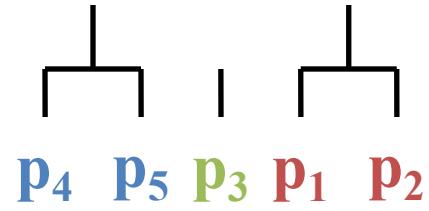
Sample 5

Sample 6

In R

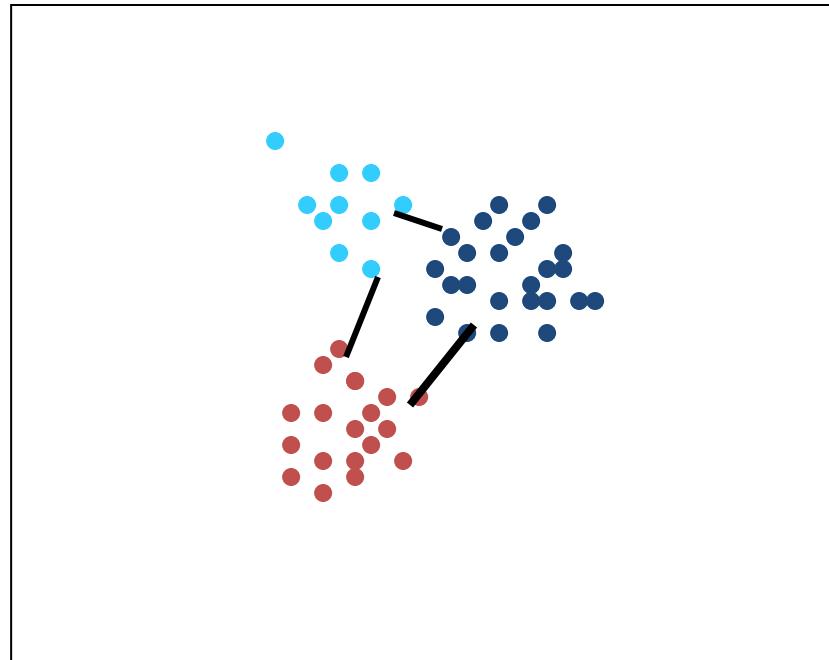
```
>?heatmap  
>heatmap(distanceMatrix, Colv=NA, Rowv=NA,  
scale="none")
```

How to aggregate clusters?



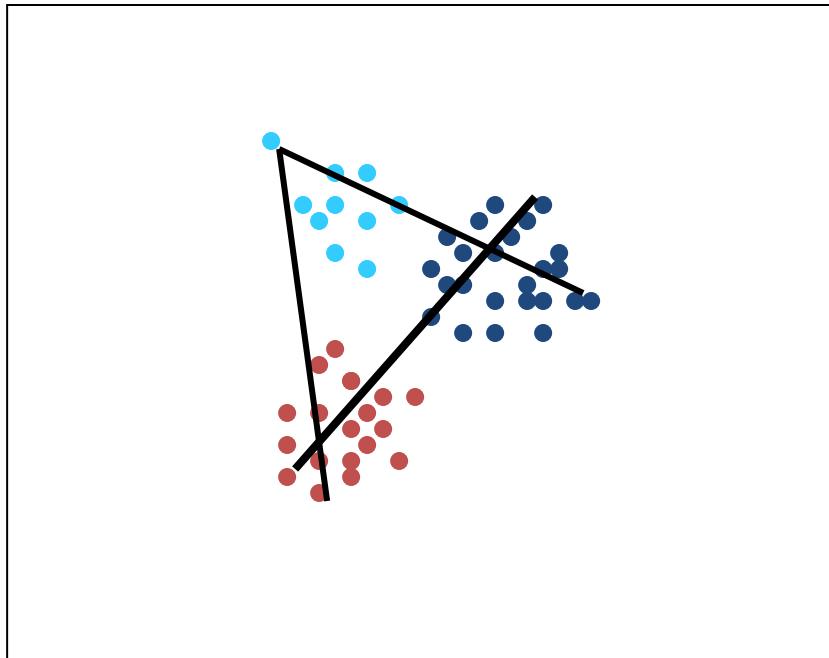
Which clusters to combine?

Single linkage



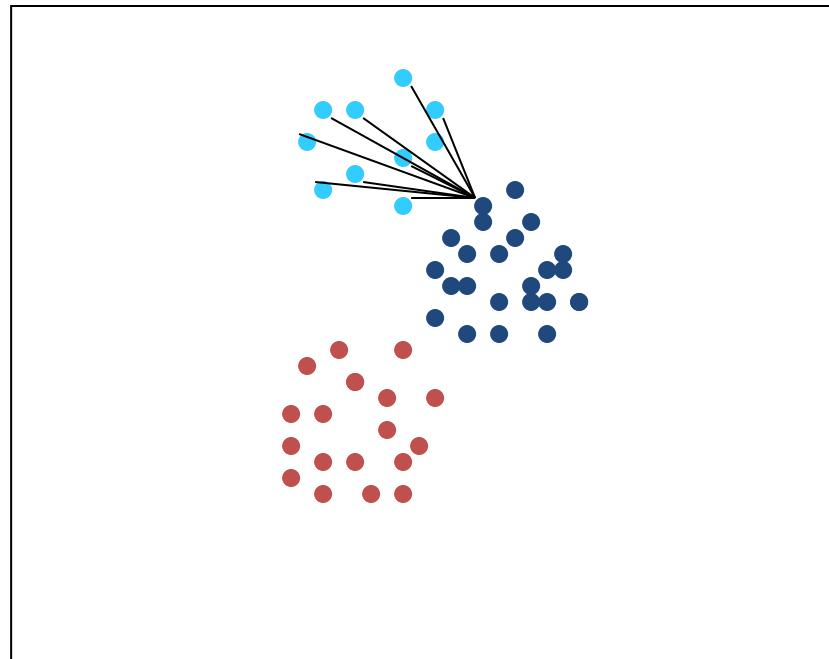
Distance between closest elements in clusters

Complete Linkage



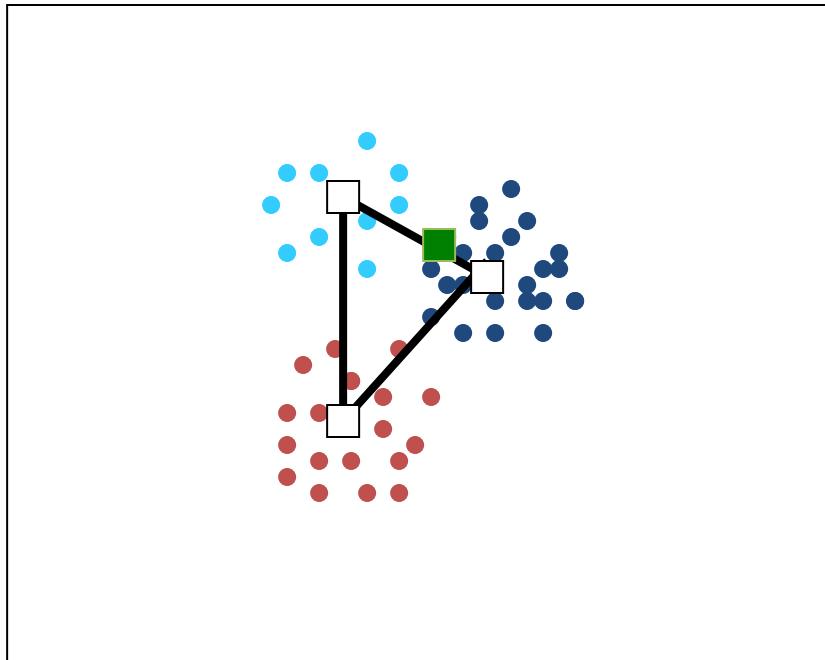
Distance between furthest elements in clusters

Average Linkage



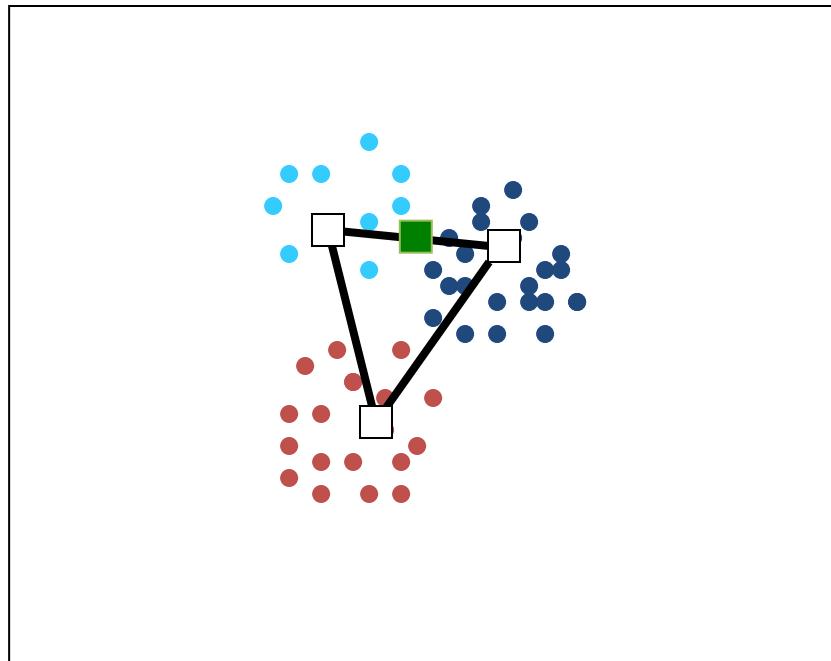
Average of all pairwise distances

Centroid Condensation (mean)



Distance between centroids (means) of two clusters

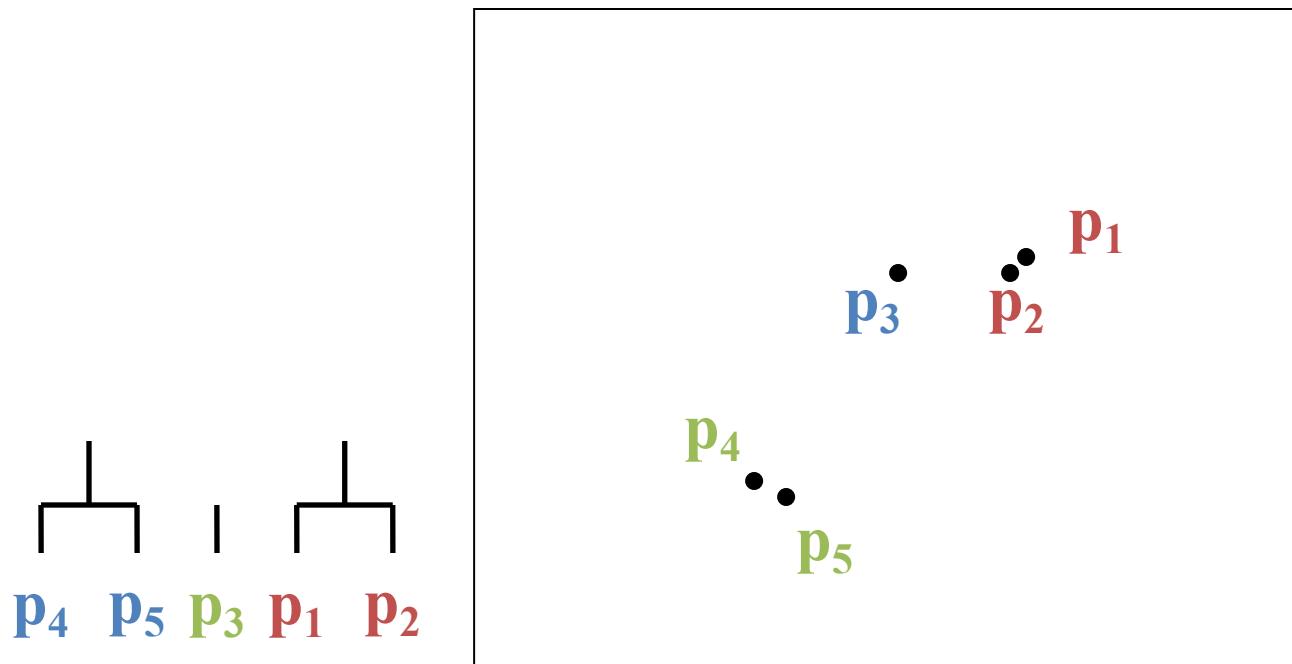
Median Condensation



Distance between median distances of two clusters

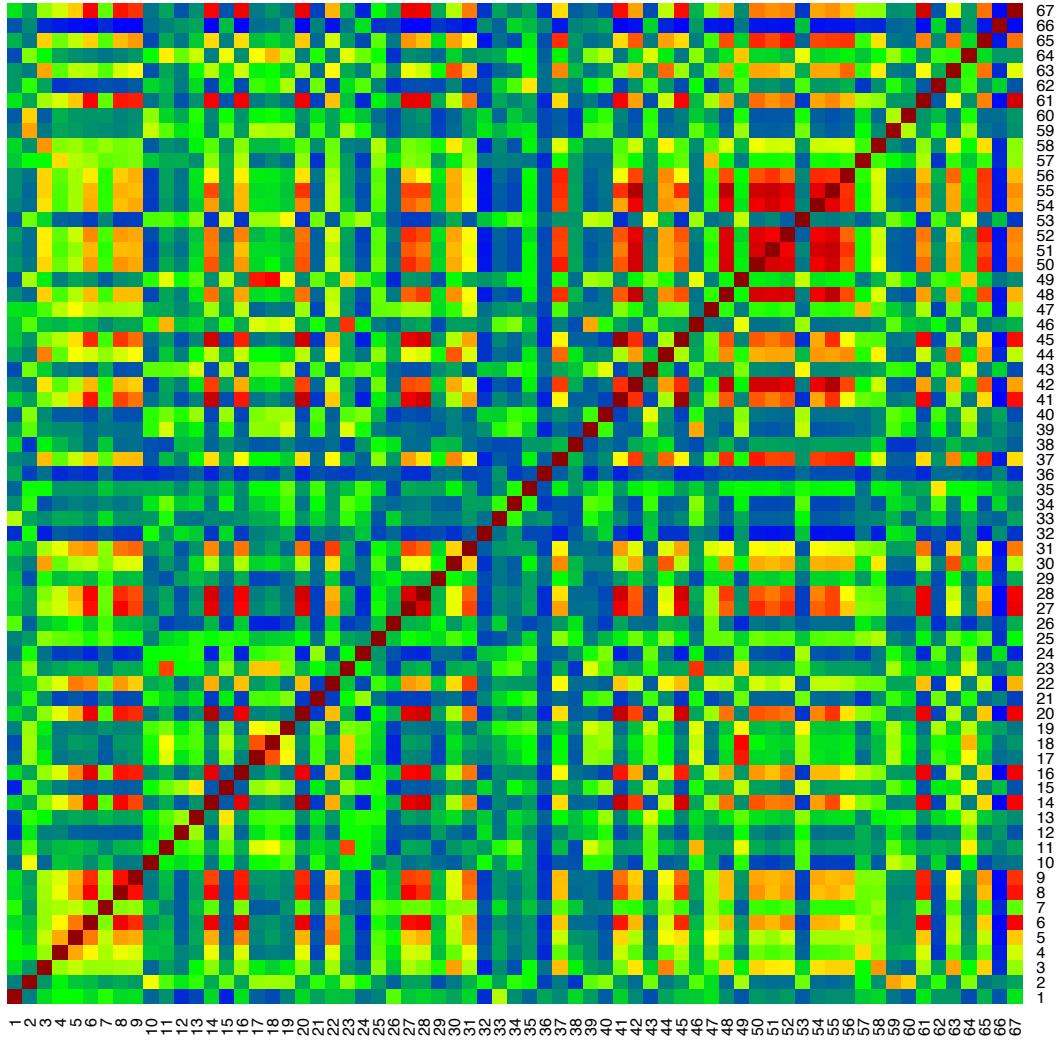
Clustering methods

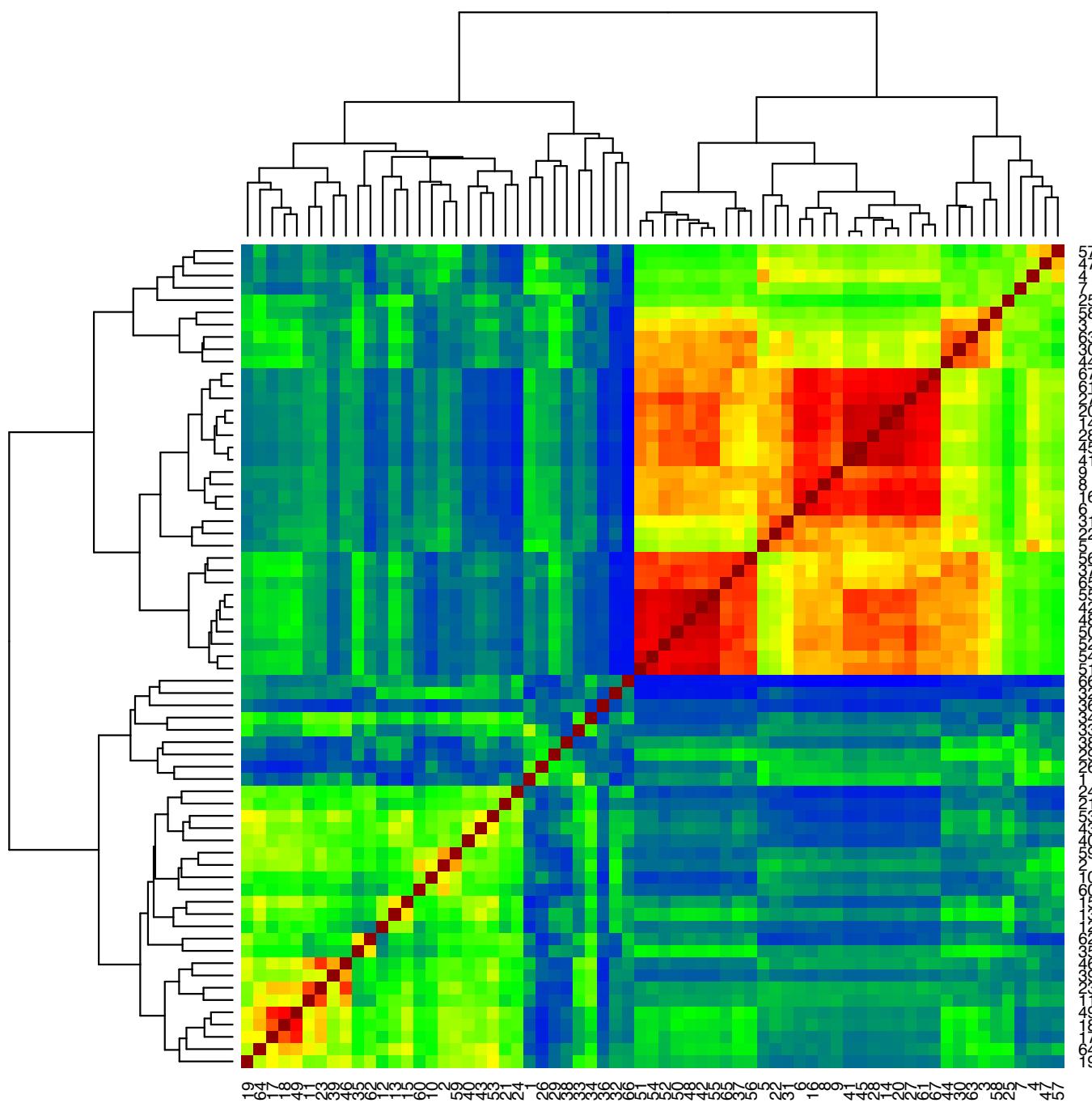
Hierarchical Clustering



At the beginning every point is a cluster in it self, then we agglomerate ...

Euclidean distance

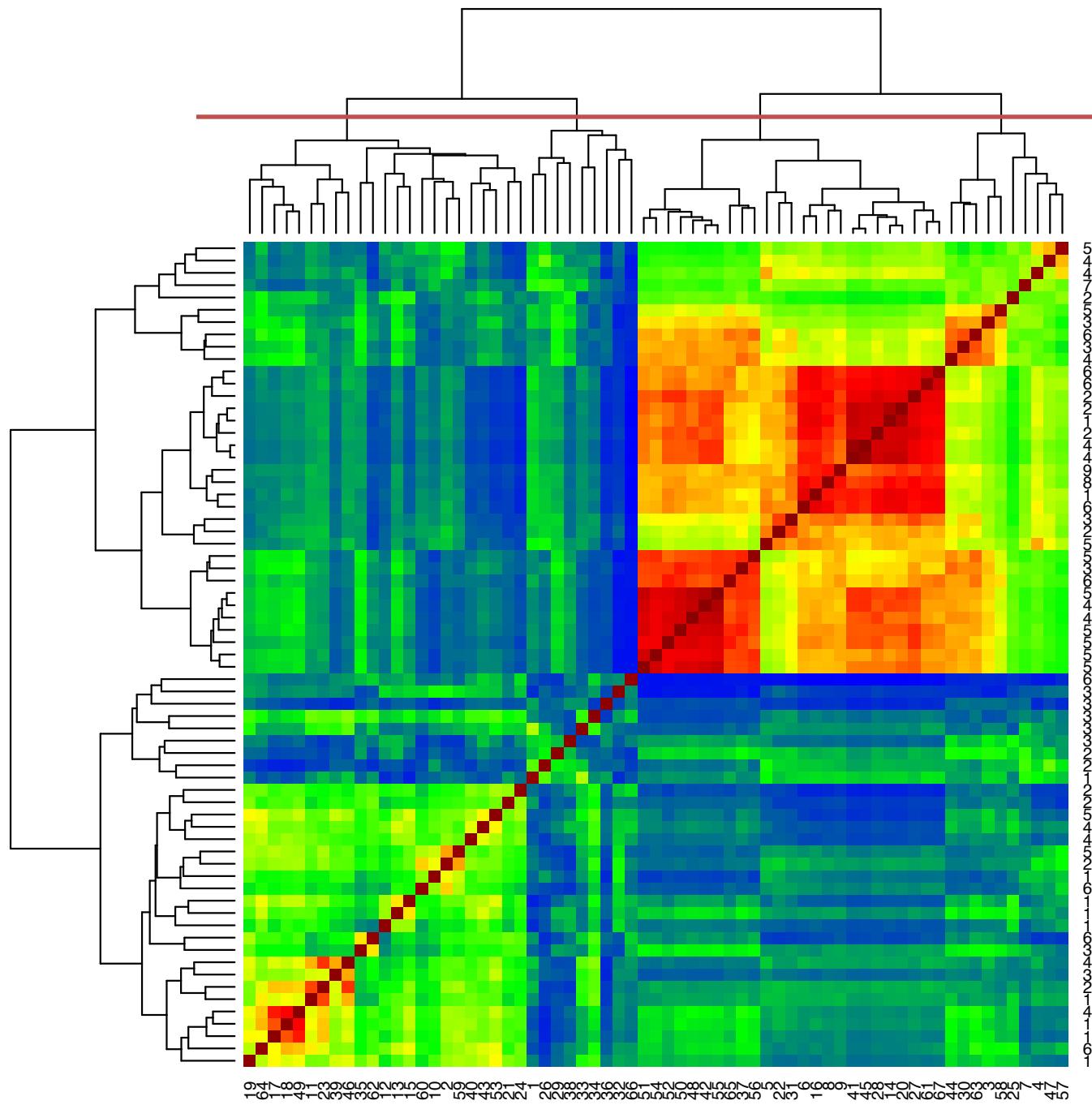




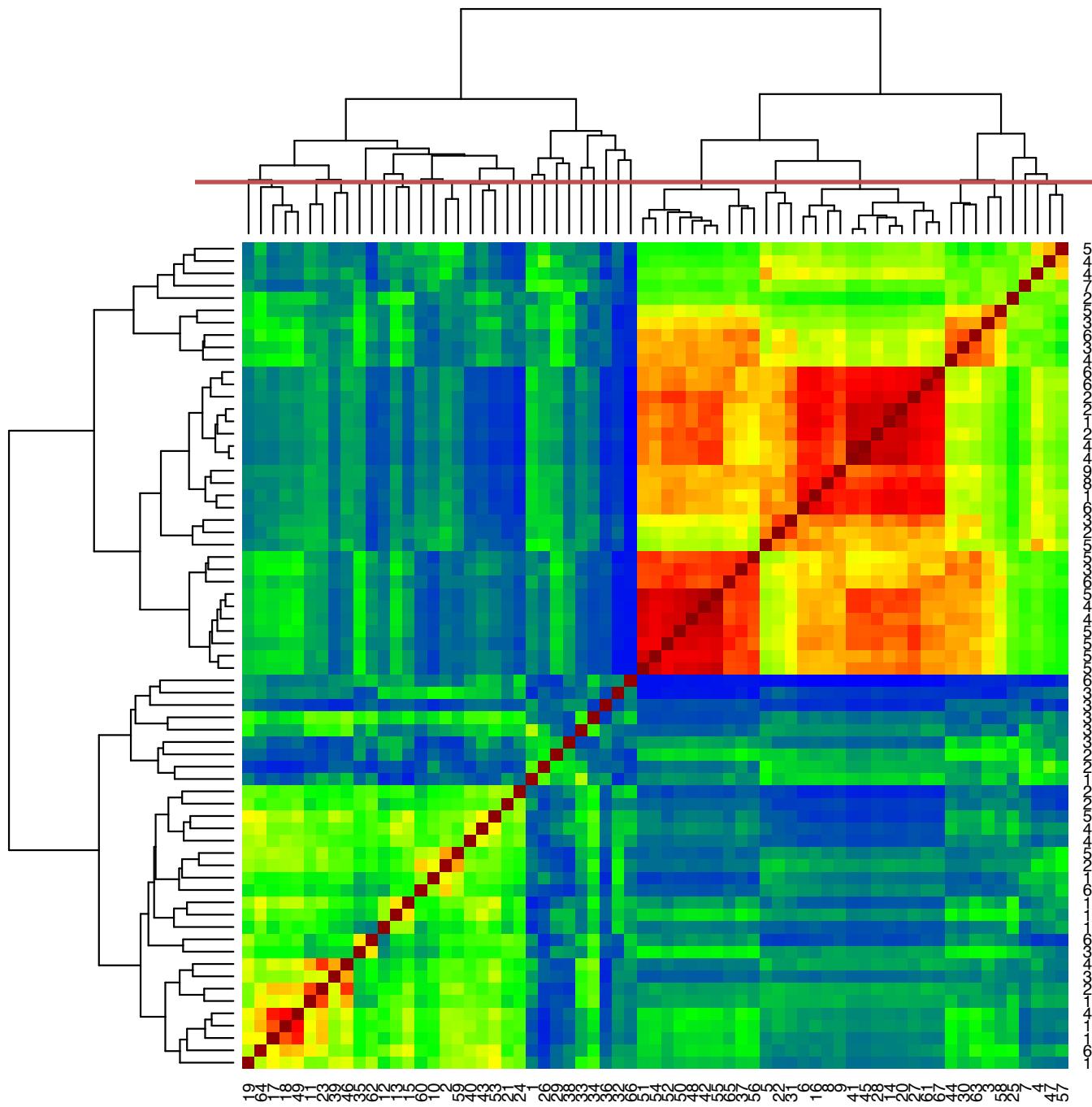
Euclidean distance
complete Linkage

Determine the Termination Condition (TC)

Euclidean distance
complete Linkage



Euclidean distance
complete Linkage



Let's practice In R

#create a random matrix

```
>mat <- matrix(data = rnorm(300, mean= 100,  
sd=10), nrow = 150, ncol = 2)
```

#Euclidian distance

```
>mat.dist<-as.matrix(dist(mat))
```

#show heatmap

```
>heatmap(mat.dist, Colv=NA, Rowv=NA, scale="none")
```

#change heatmap's color

```
>colorScale <- colorRampPalette(c("blue",  
"green", "yellow", "red", "darkred"))(1000)  
>heatmap(mat.dist, Colv=NA, Rowv=NA, scale="none",  
col=colorScale)
```

Let's practice In R

```
#create a random matrix
```

```
>mat <- matrix(data = rnorm(300, mean= 100,  
sd=10), nrow = 150, ncol = 2)
```

```
#Euclidian distance
```

```
>mat.dist<-as.matrix(dist(mat))
```

```
#show heatmap
```

```
>heatmap(mat.dist, Colv=NA, Rowv=NA, scale="none")
```

```
#change heatmap's color
```

```
>colorScale <- colorRampPalette(c("blue",
```

```
"green", "yellow", "red", "darkred"))(1000)
```

```
>heatmap(mat.dist, Colv=NA, Rowv=NA, scale="none",  
col=colorScale)
```

Let's practice In R

```
#create a random matrix
```

```
>mat <- matrix(data = rnorm(300, mean= 100,  
sd=10), nrow = 150, ncol = 2)
```

```
#Euclidian distance
```

```
>mat.dist<-as.matrix(dist(mat))
```

```
#show heatmap
```

```
>heatmap(mat.dist, Colv=NA, Rowv=NA, scale="none")
```

```
#change heatmap's color
```

```
>colorScale <- colorRampPalette(c("blue",
```

```
"green", "yellow", "red", "darkred"))(1000)
```

```
>heatmap(mat.dist, Colv=NA, Rowv=NA, scale="none",  
col=colorScale)
```

Let's practice In R

#create a random matrix

```
>mat <- matrix(data = rnorm(300, mean= 100,  
sd=10), nrow = 150, ncol = 2)
```

#Euclidian distance

```
>mat.dist<-as.matrix(dist(mat))
```

#show heatmap

```
>heatmap(mat.dist, Colv=NA, Rowv=NA, scale="none")
```

#change heatmap's color

```
>colorScale <- colorRampPalette(c("blue",  
"green", "yellow", "red", "darkred"))(1000)  
>heatmap(mat.dist, Colv=NA, Rowv=NA, scale="none",  
col=colorScale)
```

Let's practice In R

#create a random matrix

```
>mat <- matrix(data = rnorm(300, mean= 100,  
sd=10), nrow = 150, ncol = 2)
```

#Euclidian distance

```
>mat.dist<-as.matrix(dist(mat))
```

#show heatmap

```
>heatmap(mat.dist, Colv=NA, Rowv=NA, scale="none")
```

#change heatmap's color

```
>colorScale <- colorRampPalette(c("blue",  
"green","yellow","red","darkred"))(1000)  
>heatmap(mat.dist, Colv=NA, Rowv=NA, scale="none",  
col=colorScale)
```

How to do hierarchical clustering in R?

```
>?hclust
```

```
#Euclidian distance
```

```
>distE<-dist(mat)
>mat.distE<-as.matrix(dist(mat))
>heatmap(mat.distE, Colv=NA, Rowv=NA, scale="none")
>hE<-hclust(distE,"complete")
>plot(hE)
```

```
#manhattan distance
```

```
>distC<-dist(mat,method="manhattan"
>mat.distC<-as.matrix(dist(mat,method="manhattan"))
>heatmap(mat.distC, Colv=NA, Rowv=NA, scale="none")
>hC<-hclust(distC,"complete")
>plot(hC)
```

How to do hierarchical clustering in R?

```
>?hclust
```

#Euclidian distance

```
>distE<-dist(mat)
>mat.distE<-as.matrix(dist(mat))
>heatmap(mat.distE, Colv=NA, Rowv=NA, scale="none")
>hE<-hclust(distE,"complete")
>plot(hE)
```

#manhattan distance

```
>distC<-dist(mat,method="manhattan")
>mat.distC<-as.matrix(dist(mat,method="manhattan"))
>heatmap(mat.distC, Colv=NA, Rowv=NA, scale="none")
>hC<-hclust(distC,"complete")
>plot(hC)
```

How to do hierarchical clustering in R?

```
>?hclust
```

#Euclidian distance

```
>distE<-dist(mat)
>mat.distE<-as.matrix(dist(mat))
>heatmap(mat.distE, Colv=NA, Rowv=NA, scale="none")
>hE<-hclust(distE,"complete")
>plot(hE)
```

#manhattan distance

```
>distC<-dist(mat,method="manhattan")
>mat.distC<-as.matrix(dist(mat,method="manhattan"))
>heatmap(mat.distC, Colv=NA, Rowv=NA, scale="none")
>hC<-hclust(distC,"complete")
>plot(hC)
```

How to do hierarchical clustering in R?

```
>?hclust
```

#Euclidian distance

```
>distE<-dist(mat)
>mat.distE<-as.matrix(dist(mat))
>heatmap(mat.distE, Colv=NA, Rowv=NA, scale="none")
>hE<-hclust(distE,"complete")
>plot(hE)
```

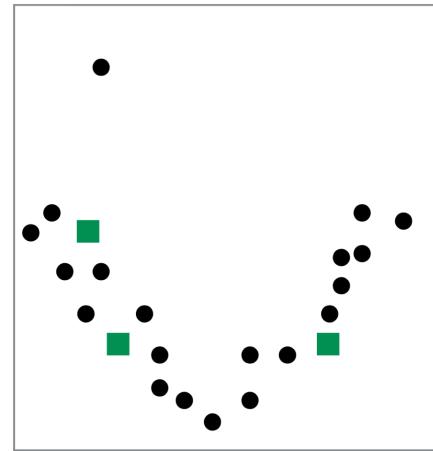
#manhattan distance

```
>distC<-dist(mat,method="manhattan")
>mat.distC<-as.matrix(dist(mat,method="manhattan"))
>heatmap(mat.distC, Colv=NA, Rowv=NA, scale="none")
>hC<-hclust(distC,"complete")
>plot(hC)
```

K-means Clustering

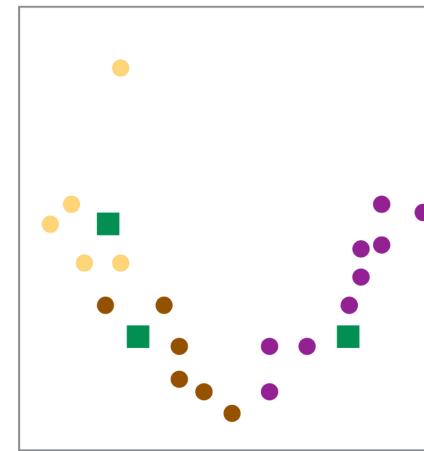
Number of clusters = 3

a



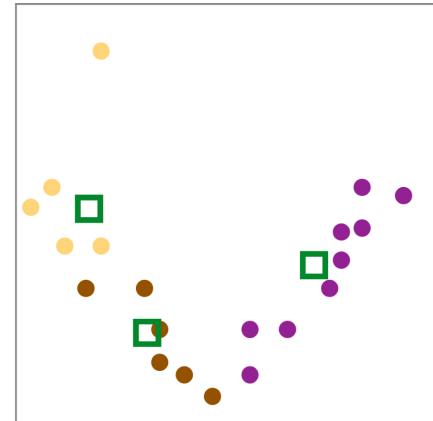
Start with
3 initial
points

b



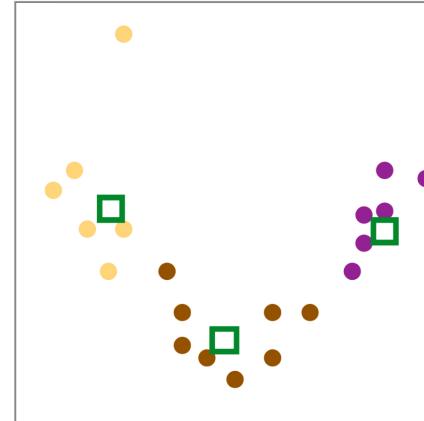
For each
point
determine to
which initial
point it is the
closest

c



Move initial
points to the
centroids of
the clusters

d

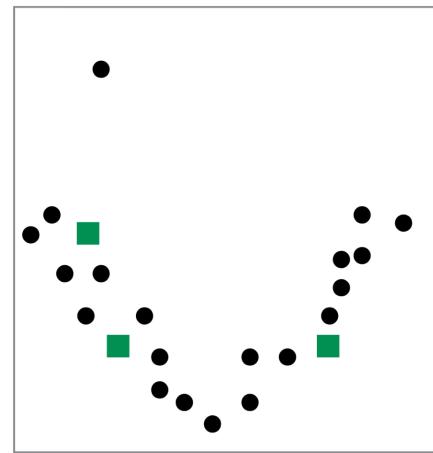


Color again
each point!
Repeat b and
c until
obtaining
stabilisation

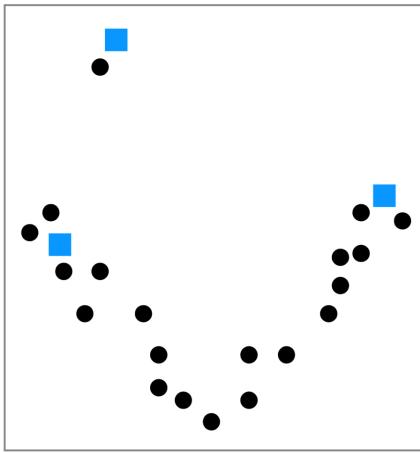
K-means Clustering

Number of clusters = 3

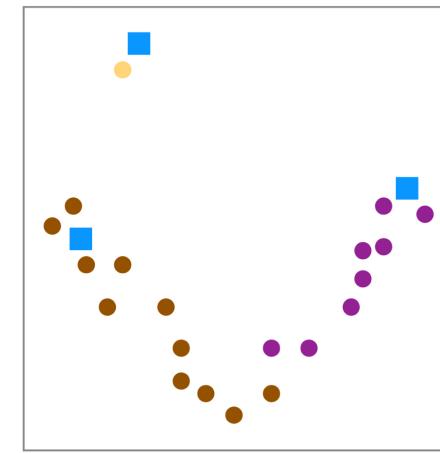
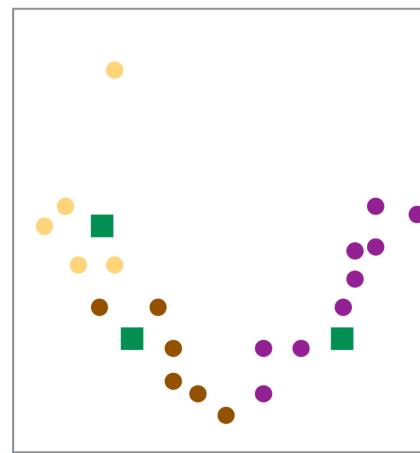
a



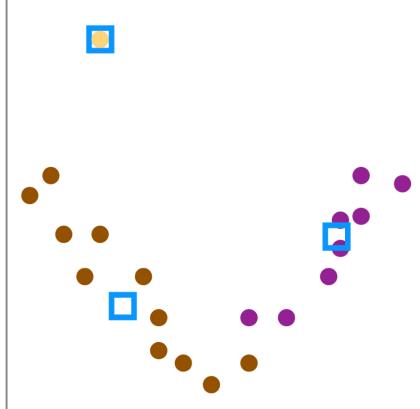
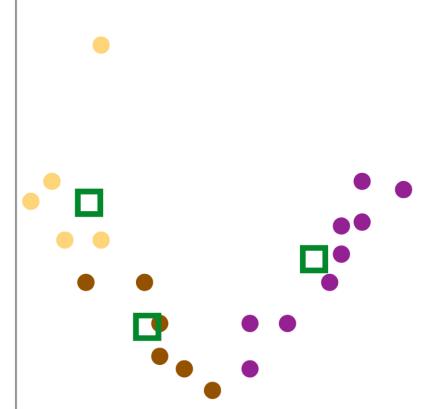
b



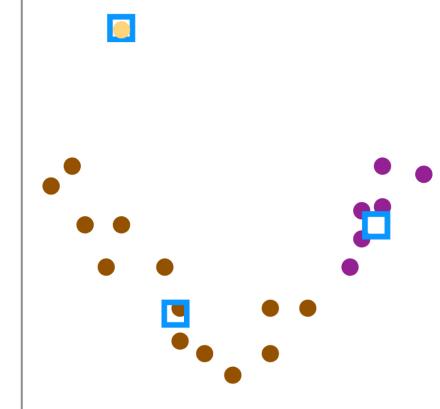
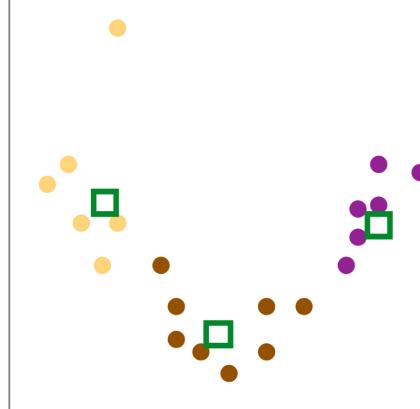
b



c



d



K-means & C-means

Drawbacks:

1. Specify number of clusters
2. Non probabilistic methods
3. Not stable

Kmeans in R

```
>mat <- matrix(data = rnorm(300, mean=100, sd=10),  
                 nrow = 150,  
                 ncol = 2)  
>df<-data.frame(x)  
  
>kmeans(df, 3)
```

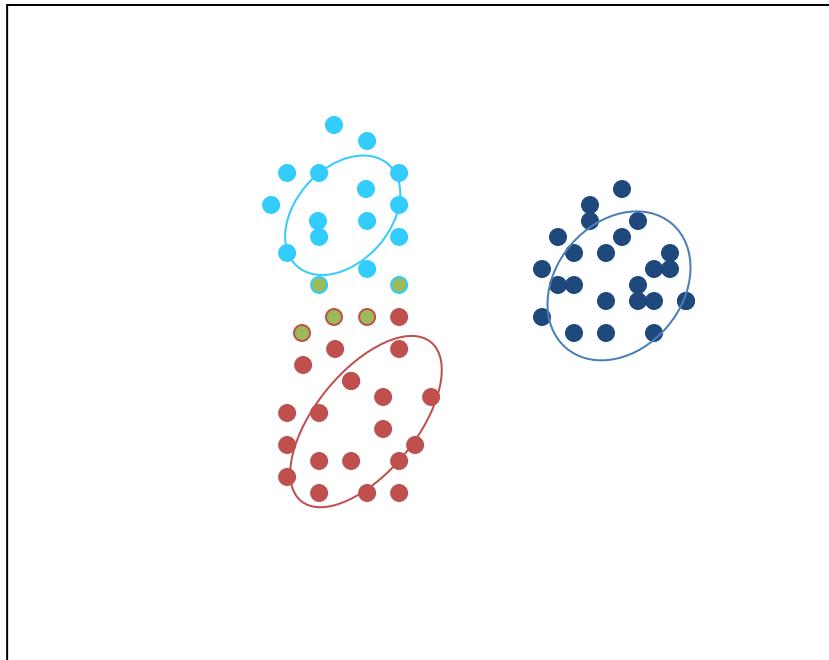
Kmeans in R

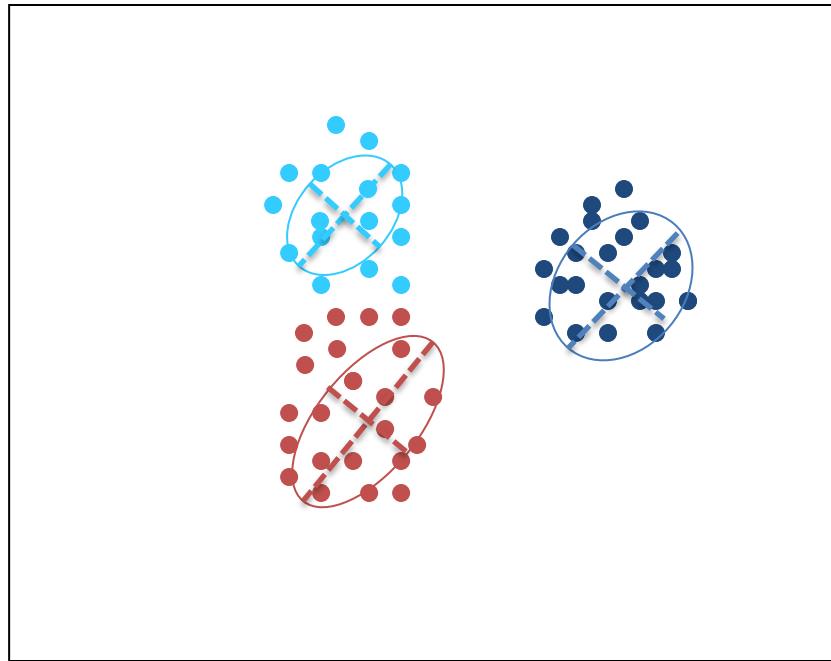
```
>mat <- matrix(data = rnorm(300, mean= 100, sd=10),  
                 nrow = 150,  
                 ncol = 2)  
>df<-data.frame(mat)  
  
>kmeans(df, 3)  
  
>cl.1 <- kmeans(df, 3, iter.max = 1)  
>plot(df, col = cl.1$cluster)  
>points(cl.1$centers, col = 1:5, pch = 8)
```

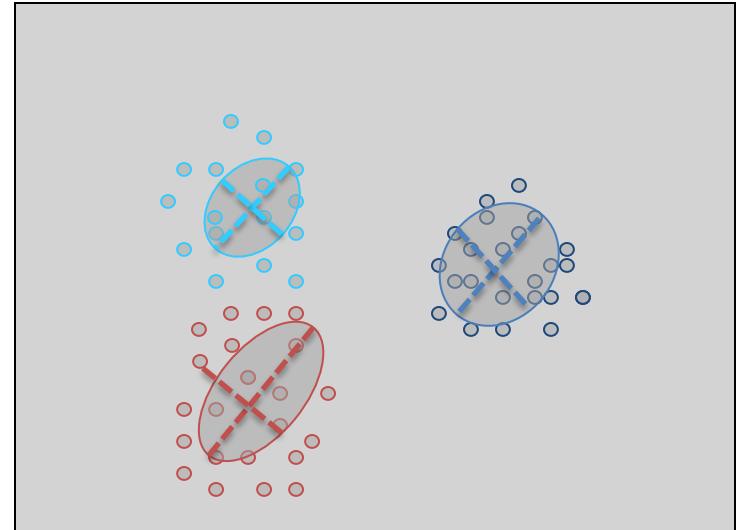
Kmeans in R

```
>mat <- matrix(data = rnorm(300, mean= 100, sd=10),  
                 nrow = 150,  
                 ncol = 2)  
>df<-data.frame(mat)  
  
>kmeans(df, 3)  
  
>cl.1 <- kmeans(df, 3, iter.max = 1)  
>plot(df, col = cl.1$cluster)  
>points(cl.1$centers, col = 1:5, pch = 8)  
  
>cl.10 <- kmeans(df, 3, iter.max = 10)  
>plot(df, col = cl.10$cluster)  
>points(cl.10$centers, col = 1:5, pch = 8)  
  
>cl.100 <- kmeans(df, 3, iter.max = 100)  
>plot(df, col = cl.100$cluster)  
>points(cl.100$centers, col = 1:5, pch = 8)
```

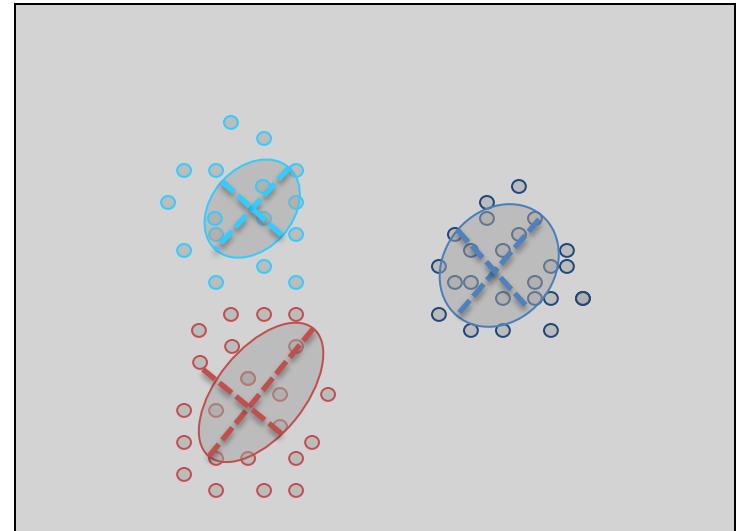
Model-based Clustering



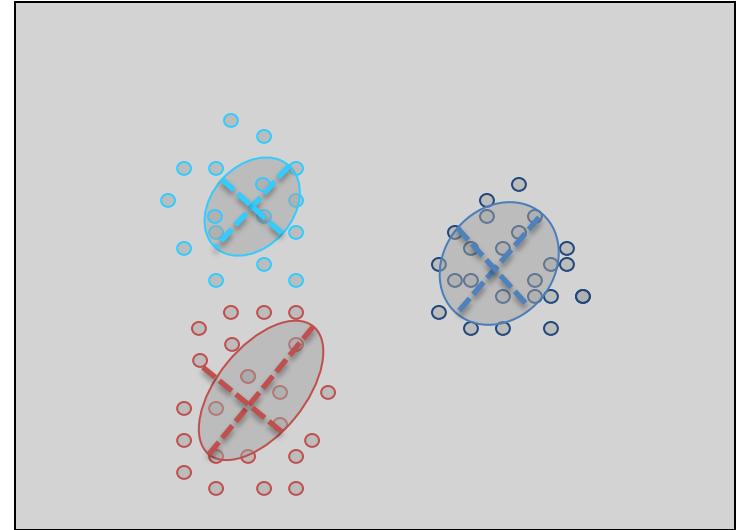




distribution
(univariate, spherical, diagonal,
ellipsoidal)

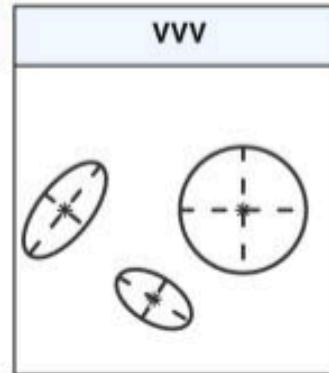
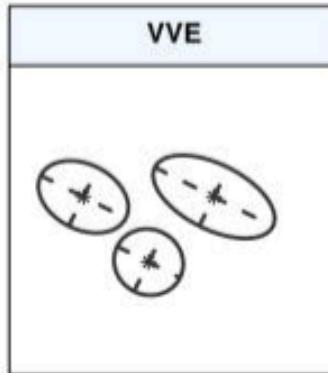
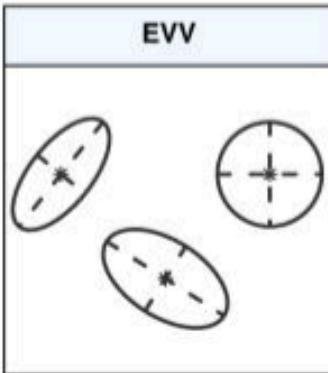
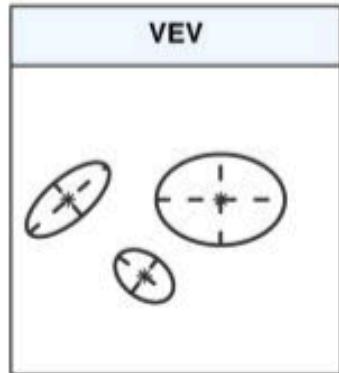
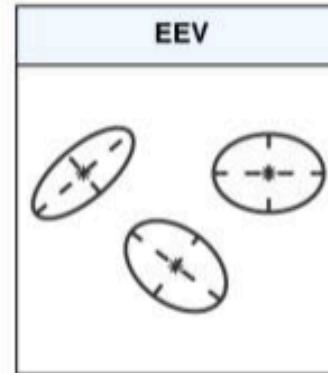
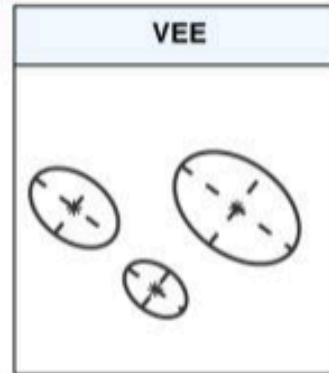
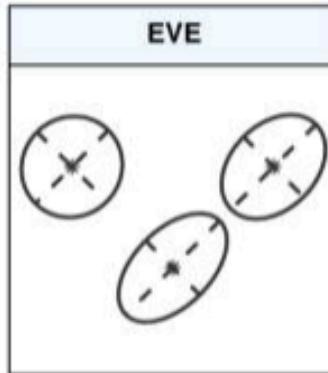
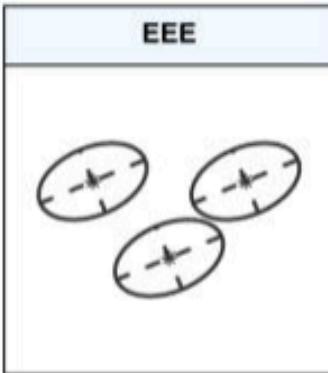
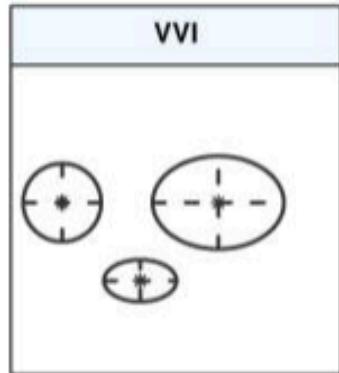
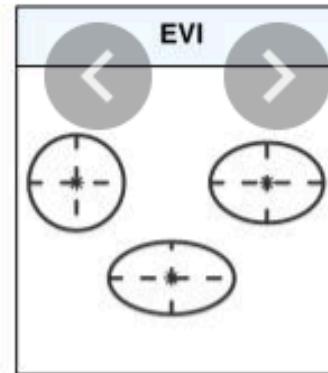
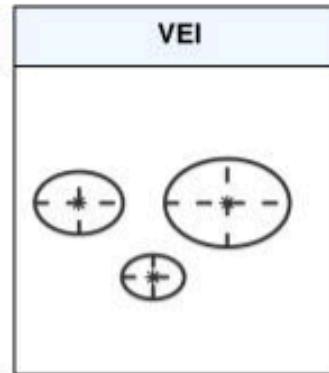
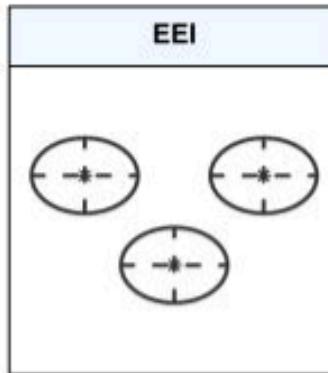
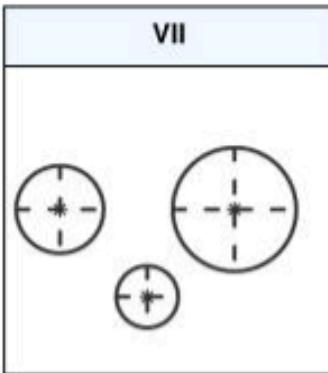
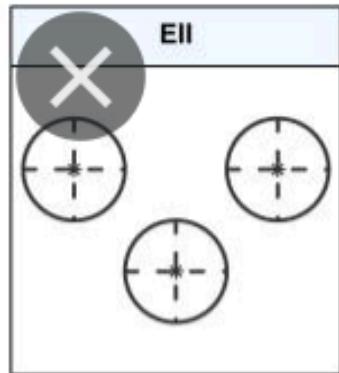


data volume
(equal, variable)



shape
(equal, variable)

Volume, Shape, Orientation



Model selection

identifier	Model	HC	EM	Distribution	Volume	Shape	Orientation
E		•	•	(univariate)	equal		
V		•	•	(univariate)	variable		
EII	λI	•	•	Spherical	equal	equal	NA
VII	$\lambda_k I$	•	•	Spherical	variable	equal	NA
EEI	λA		•	Diagonal	equal	equal	coordinate axes
VEI	$\lambda_k A$		•	Diagonal	variable	equal	coordinate axes
EVI	λA_k		•	Diagonal	equal	variable	coordinate axes
VVI	$\lambda_k A_k$		•	Diagonal	variable	variable	coordinate axes
EEE	λDAD^T	•	•	Ellipsoidal	equal	equal	equal
EEV	$\lambda D_k AD_k^T$		•	Ellipsoidal	equal	equal	variable
VEV	$\lambda_k D_k AD_k^T$		•	Ellipsoidal	variable	equal	variable
VVV	$\lambda_k D_k A_k D_k^T$	•	•	Ellipsoidal	variable	variable	variable

BIC= Bayesian information criterion

BIC

- the model with **the highest BIC** is preferred
- BIC is a function of the number of parameters of the model
- The goodness of the fit of the model
- The sample size

Number of
parameters

Best likelihood



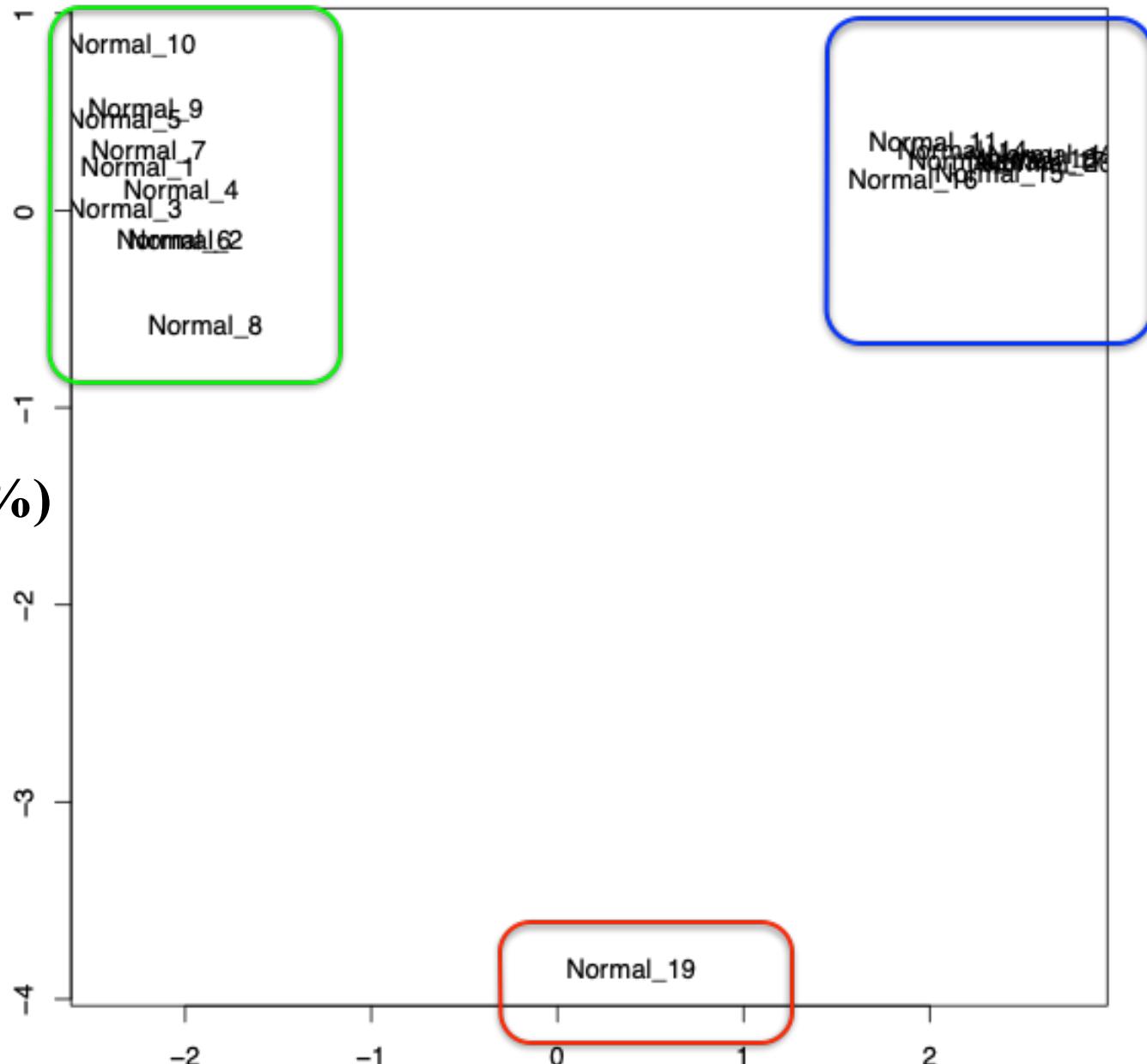
Mclust In R

```
>?mclustBIC  
>?Mclust  
  
>BIC <- mclustBIC(df)  
>plot(BIC)  
>summary(BIC)  
>mod1 <- Mclust(df, x = BIC)  
>summary(mod1, parameters = TRUE)  
>plot(mod1, what = "classification")
```

**Once you have the clusters,
what do you do with them ?**

PC2 (12.8%)

PC1 (20.5%)



Challenge

Points in plates

1. Import the data from dataClustering.csv
2. What is the dimension of this dataset?
3. How many data point do we have?
4. Evaluate Euclidean distance of points in a plates
5. Classify points to find clusters using hierarchical clustering and the average agglomeration method

Challenge

Points in plates-continuous

6. We expect to have 3 clusters. When you apply k-means algorithm using 1 iteration, does it differ from applying it using 10 or 100 iterations?

7. What is the outcome of the C-means clustering?

```
install.packages("e1071")
```

```
library(e1071)
```

```
?cmeans
```

Challenge

Points in plates-continuous

Library(mclust)

8. What are the top 3 models *mclustBIC* function suggests based on the BIC criterion?
9. How many clusters did it find using the top model?
10. Plot the outcome

Thank you for your attention