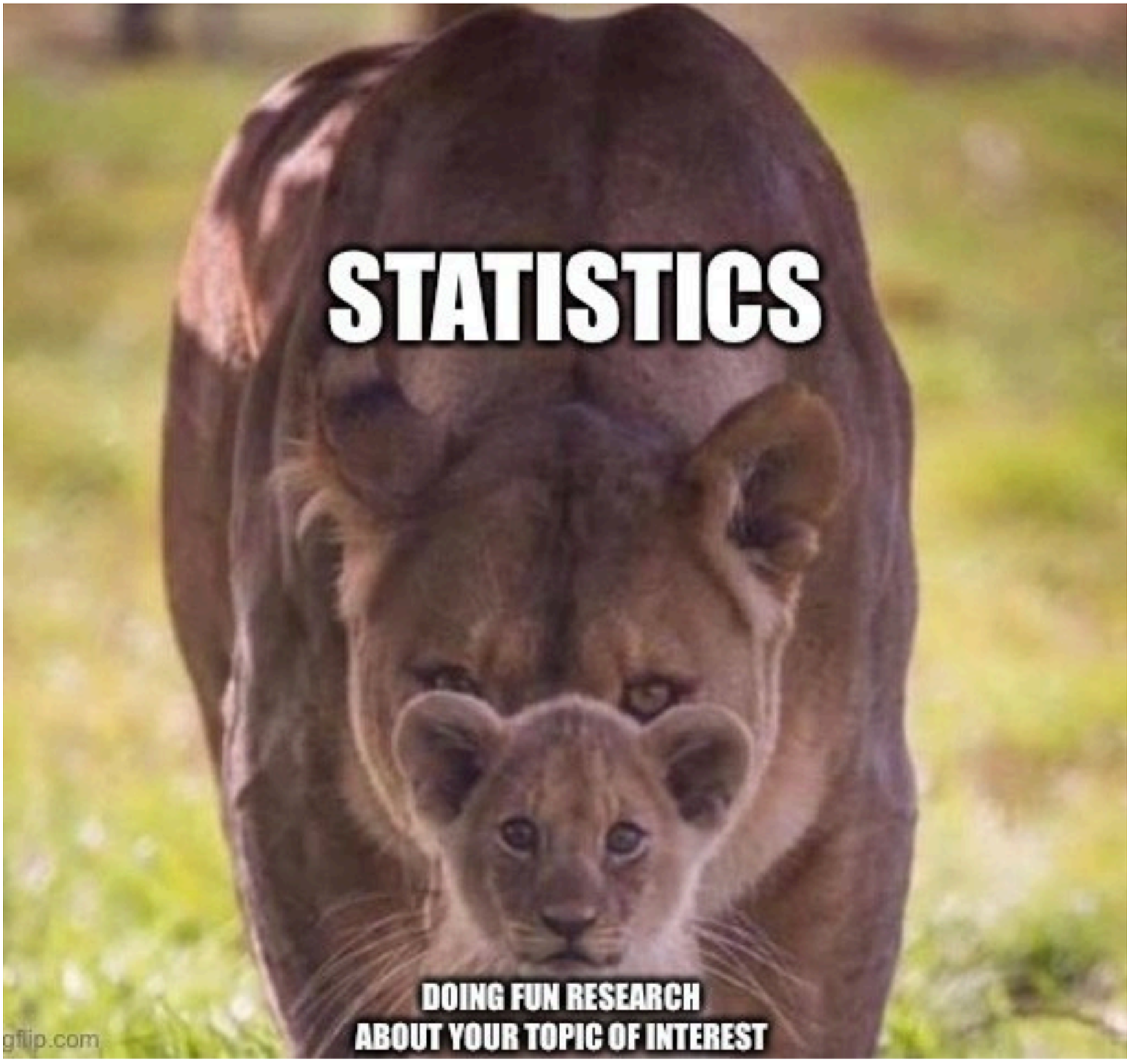
Davide Risso
Department of Statistical Sciences
davide.risso@unipd.it

FIRST MS BIOSTAT SCHOOL

STATISTICAL METHODS FOR MS DATA ANALYSIS

INTRODUCTION

WHY ARE YOU HERE?



THREE INTENSIVE DAYS

- We will try to give you a broad overview of the statistical tools needed for the analysis of large proteomics experiments.
- Unsurprisingly, we do not have enough time to go into the details of the methods.
- I will focus on intuition and examples rather than formulas and theory.
- Broadly speaking we will cover three main topics: dimensionality reduction, clustering, supervised analysis.

A COMMON THEME: HIGH DIMENSIONALITY

- The common theme of these lectures is that proteomics data are highly dimensional.
- This is one of the more challenging areas of statistics and many methods have been developed to overcome the limitation of classical statistical methods.
- There are many good resources to deepen your knowledge, I've listed my favorite (free!) ones at the end of the slides.

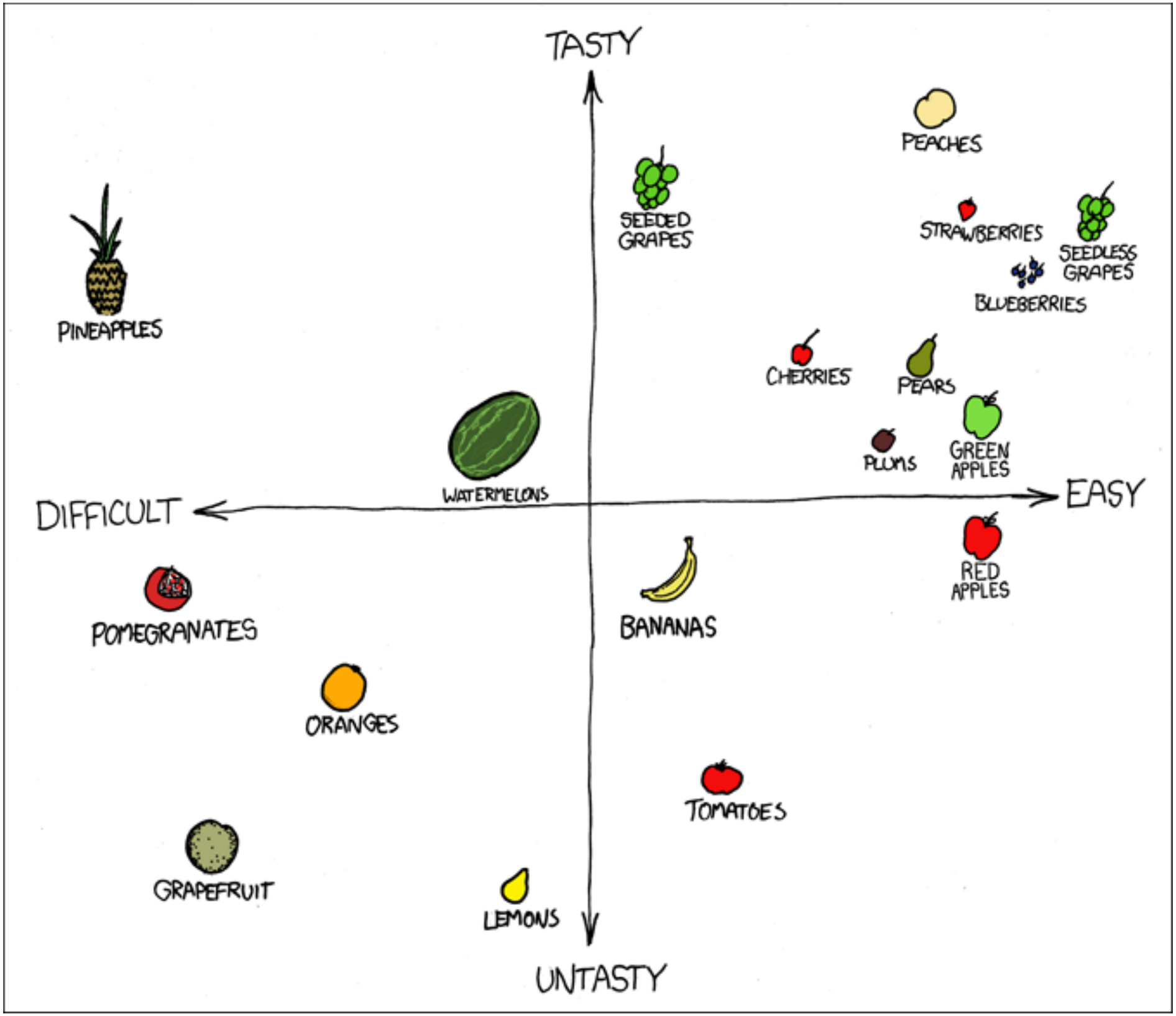
DIMENSIONALITY REDUCTION

DIMENSIONALITY REDUCTION

We talk about "dimensionality reduction" when referring to two different goals:

1. **Visualize** high-dimensional data
   - Usually 2-3 dimensions
   - PCA, t-SNE, UMAP
2. **Infer** low-rank signal from high-dimensional data
   - Usually 10-50 dimensions
   - PCA, Factor Analysis, …

WHY? INTUITION FOR HIGHLY COMPLEX DATA



WHY? THE CURSE OF DIMENSIONALITY

There are several ways to define this “curse”, but with reference to the Euclidean distance, it is intuitive to think that the space becomes so vast that distances become meaningless, as “no-one is close to anyone”.

For this reason, statistical inference is often applied after an initial **dimensionality reduction step**.

PRINCIPAL COMPONENT ANALYSIS (PCA)

- PCA is the starting point and baseline approach for both types of analysis.
- PCA can be used to visualize high-dimensional data in 2-3 dimensions.
- PCA can be seen as a solution of a factor analysis model for Gaussian data.

DIMENSIONALITY REDUCTION

- Data consist of variables recorded on observational units.
- The data for p variables and n observations can be represented as a $n \times p$ data matrix
  $X = (X_{ij} : i = 1,\ldots,n; j = 1,\ldots,p)$.
- In proteomics, often the number of variables p is larger than (or at least of the same order of) the number of observations n.

DIMENSIONALITY REDUCTION

- Dimensionality reduction, i.e., representing the data using fewer than p variables, is useful for summarizing and visualizing data, in the context of exploratory data analysis (EDA, e.g., detecting main features), quality assessment/control (QA/QC, e.g., detecting artifacts, outliers), and reporting of results (e.g., clusters).
- A variety of often related approaches can be used, but we focus on PCA.
- Principal component analysis (PCA) replaces the original variables with fewer orthogonal linear combinations of these variables, with successively maximal variance.

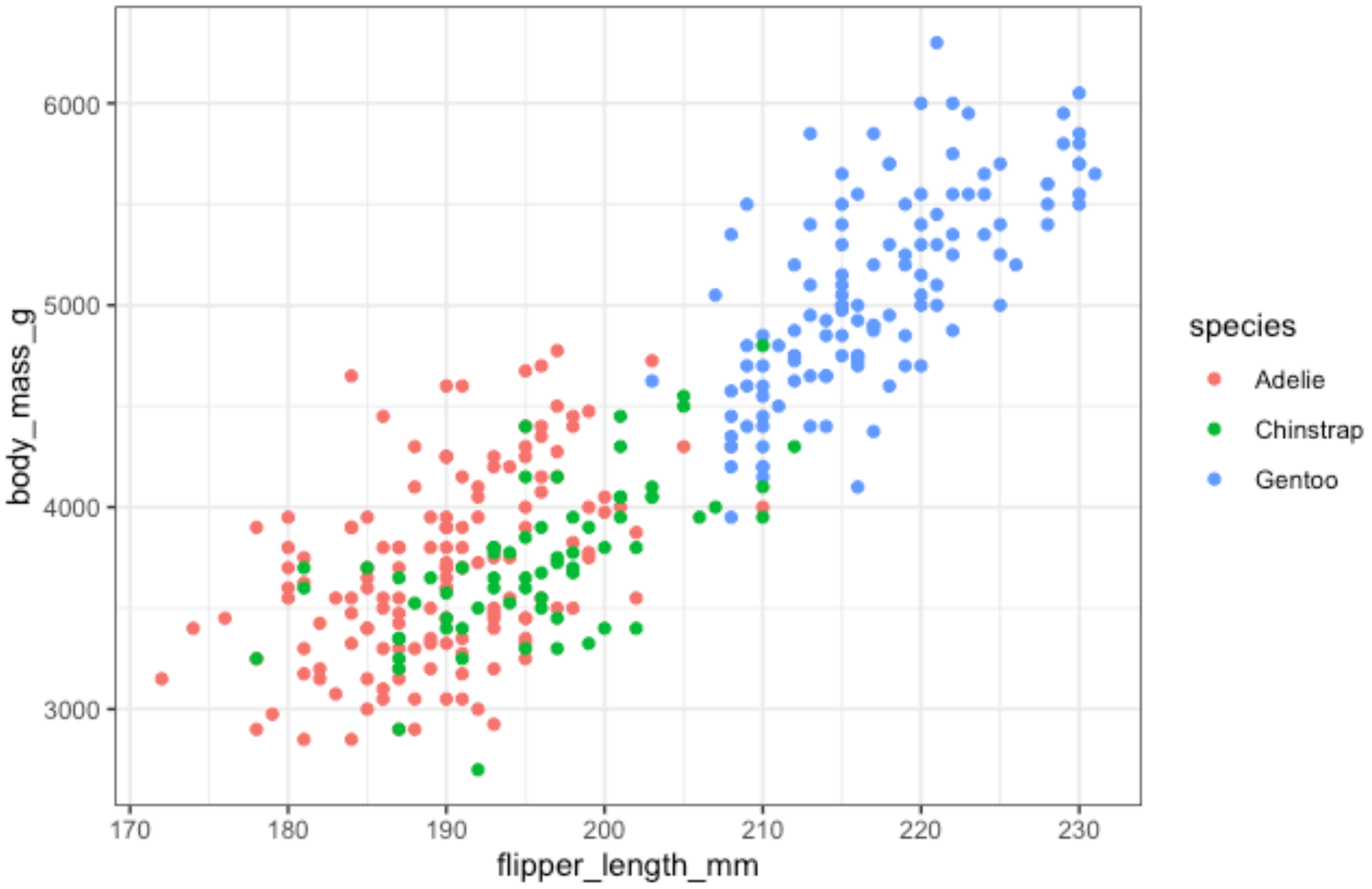
PRINCIPAL COMPONENT ANALYSIS

- Principal component analysis (PCA) is a dimensionality reduction technique that provides a parsimonious summarization of the data by replacing the original variables by fewer linear combinations of these variables, that are orthogonal and have successively maximal variance.

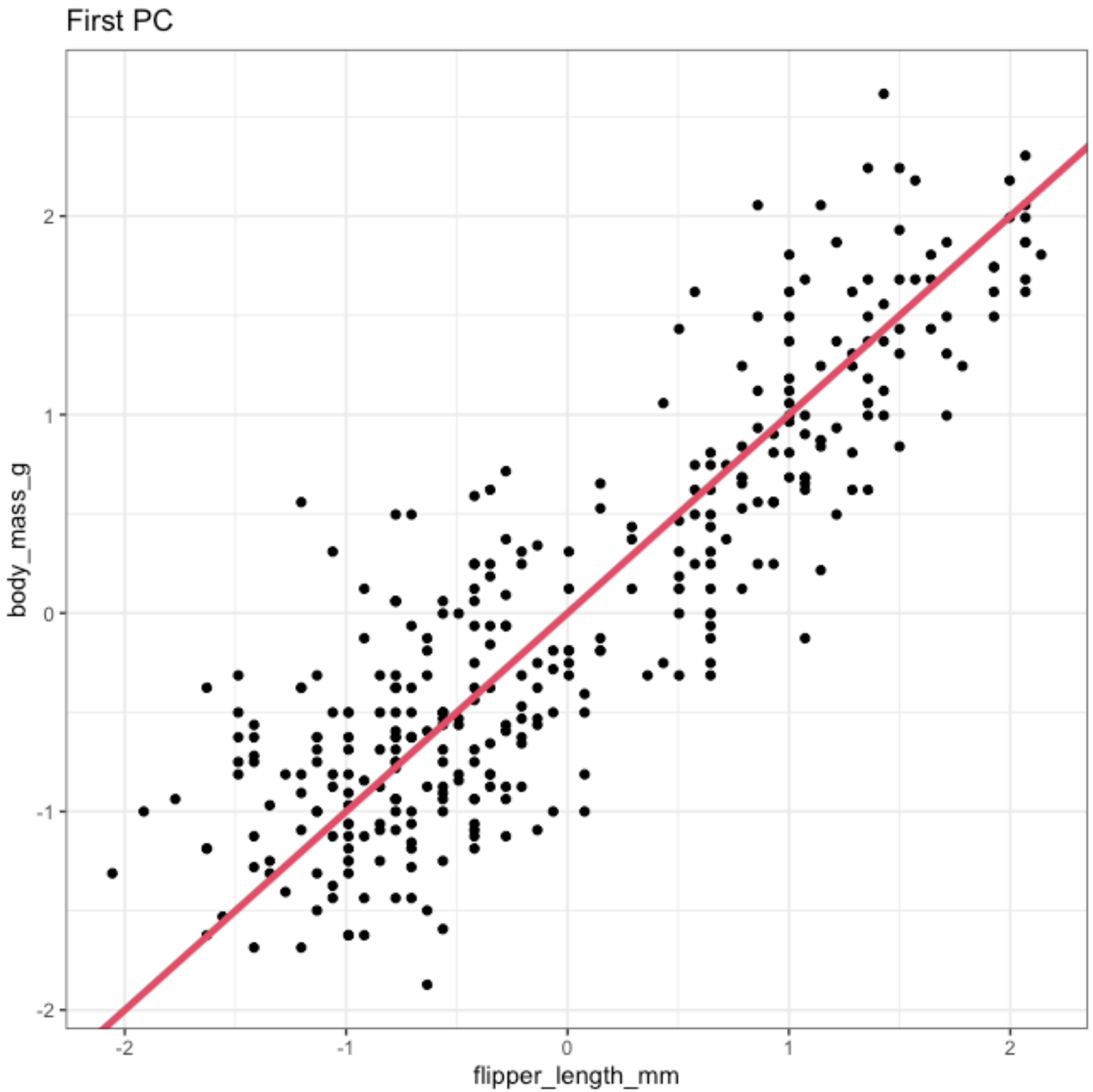
- Such linear combinations seek to “separate out” the observations, while loosing as little information as possible.

A FIRST SIMPLE EXAMPLE (UNRELATED TO PROTEOMICS)

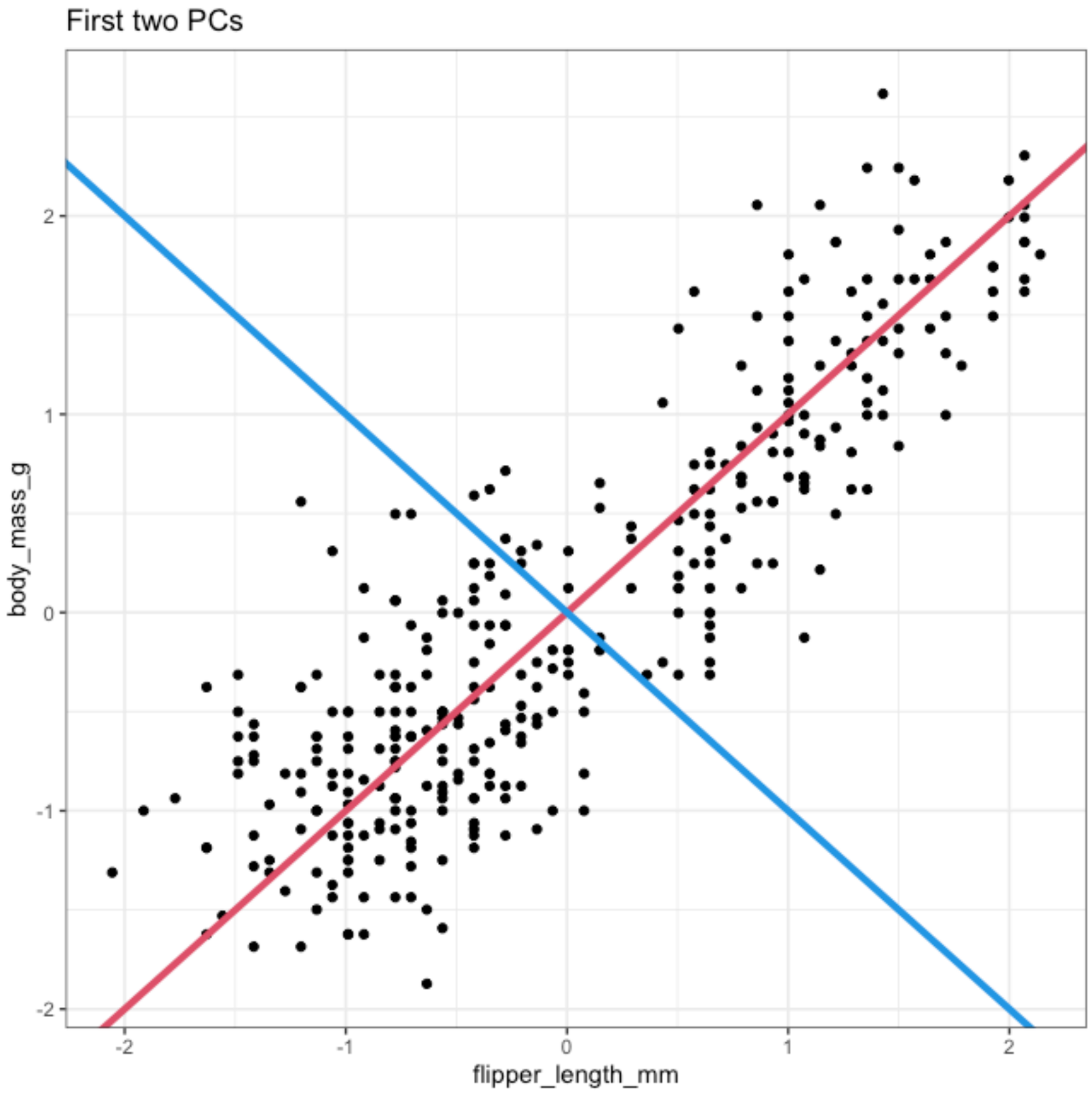
- We start with this simple example of bill and flipper measurements of some species of penguins



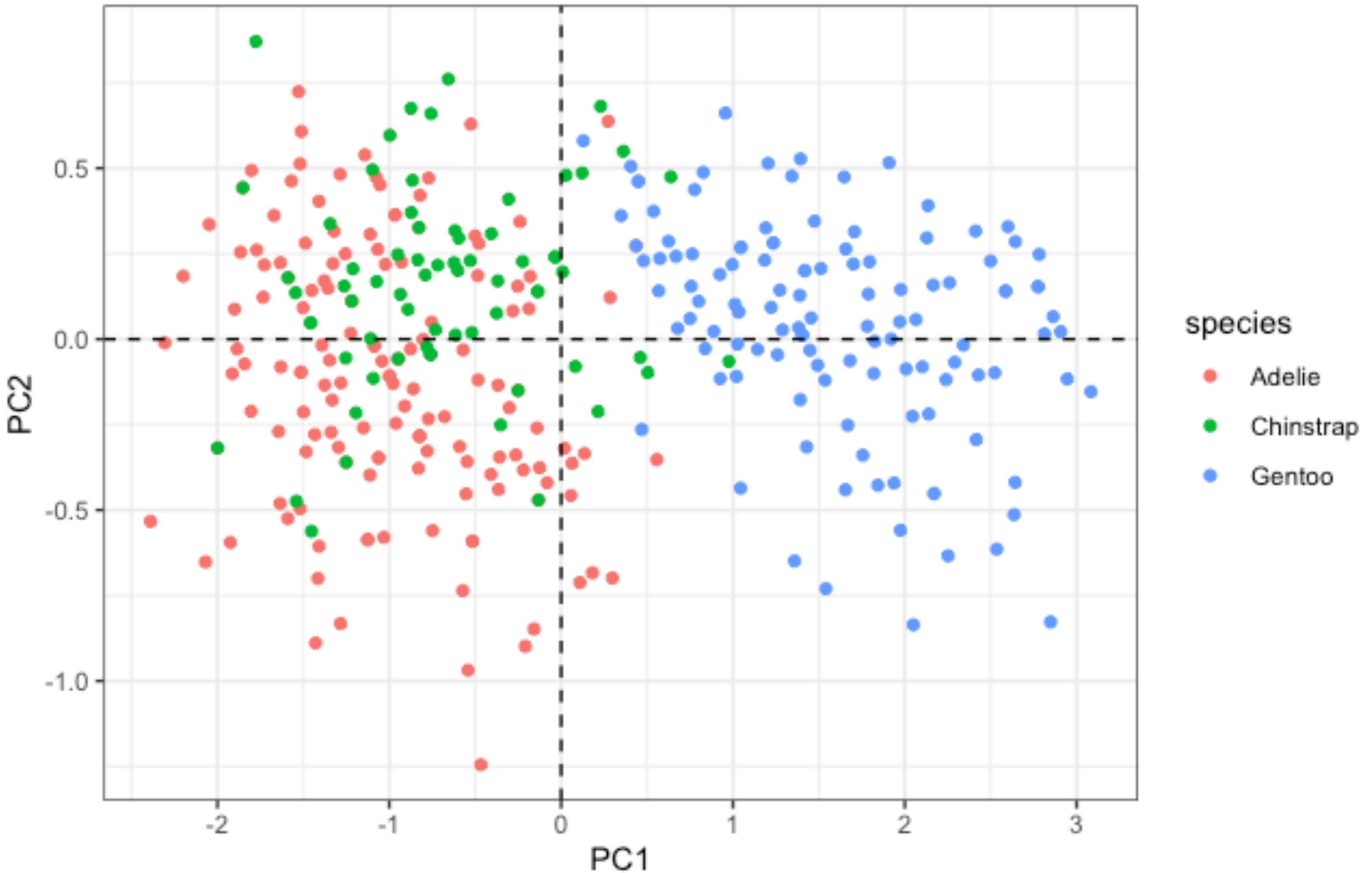
DIRECTION OF MAXIMAL VARIABILITY



ORTHOGONAL DIRECTION



LET US CHANGE THE AXIS (ROTATE THE PLOT)



DID WE REDUCE THE DIMENSION OF THE PROBLEM?

- Since the two variables are highly correlated, only one dimension (variable) carries most of the information, after a proper rotation of the data.
- By retaining only the first principal component, we reduce the dimensionality of the problem, while retaining most of the information in the data.
- Indeed, in this case the first PC "explains" 93.6% of the variability of the data.

(A LITTLE) MORE FORMALLY

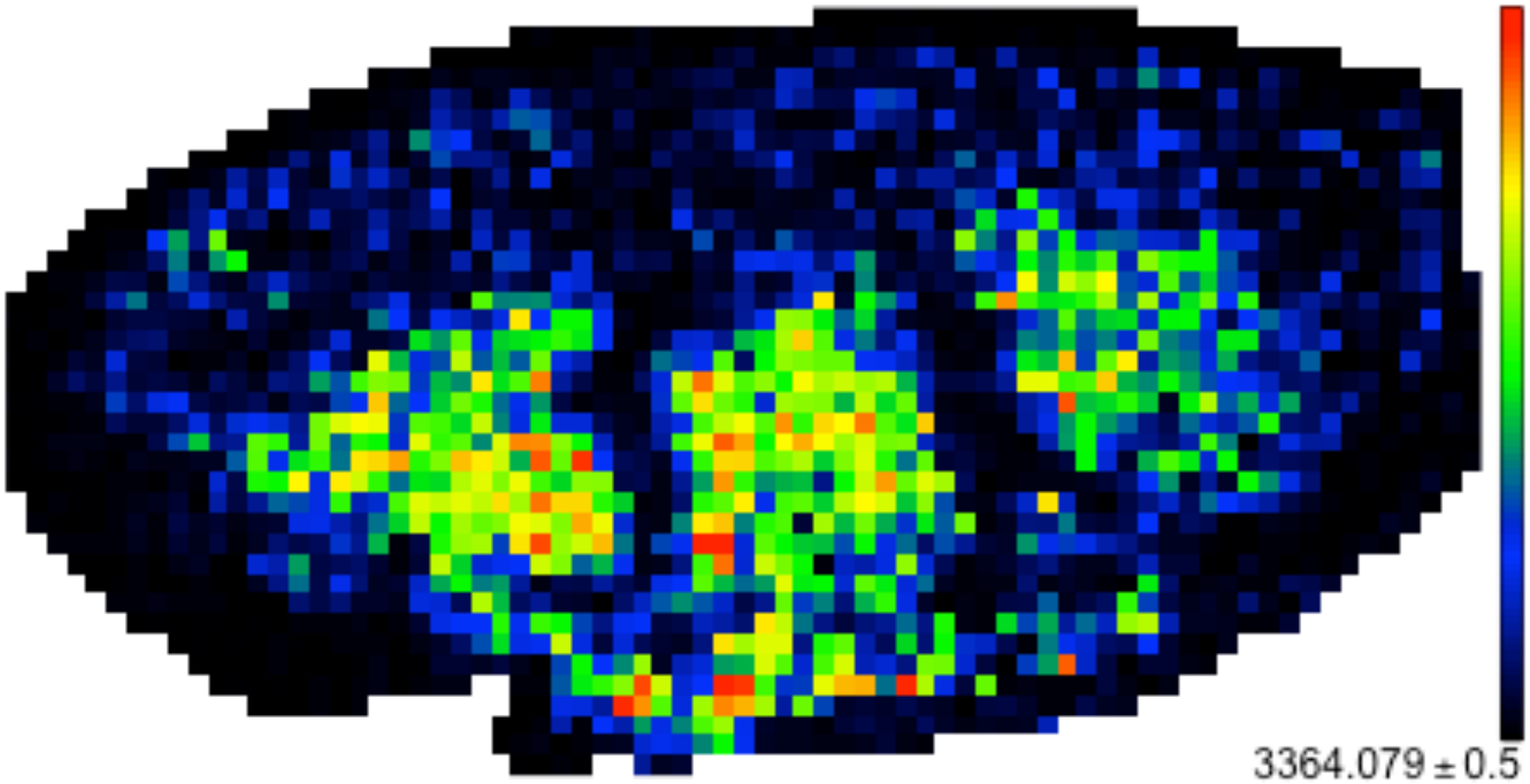
- In general we have p variables and we can compute $\min\{n, p\}$ principal components.
- The first principal component (PC1) is a linear combination of the original variables.
  $Y_1 = a_{11}X_1 + a_{12}X_2 + \ldots + a_{1p}X_p$
  whose weights $a_{11}, \ldots, a_{1p}$ are determined so that Y_1 captures most of the variability of the data.
- The weights are called *loadings*.

(A LITTLE) MORE FORMALLY

- The second principal component (PC2) is a linear combination of the original variables.
  $Y_2 = a_{21}X_1 + a_{22}X_2 + \ldots + a_{2p}X_p$
  whose weights $a_{21}, \ldots, a_{2p}$ are determined so that:

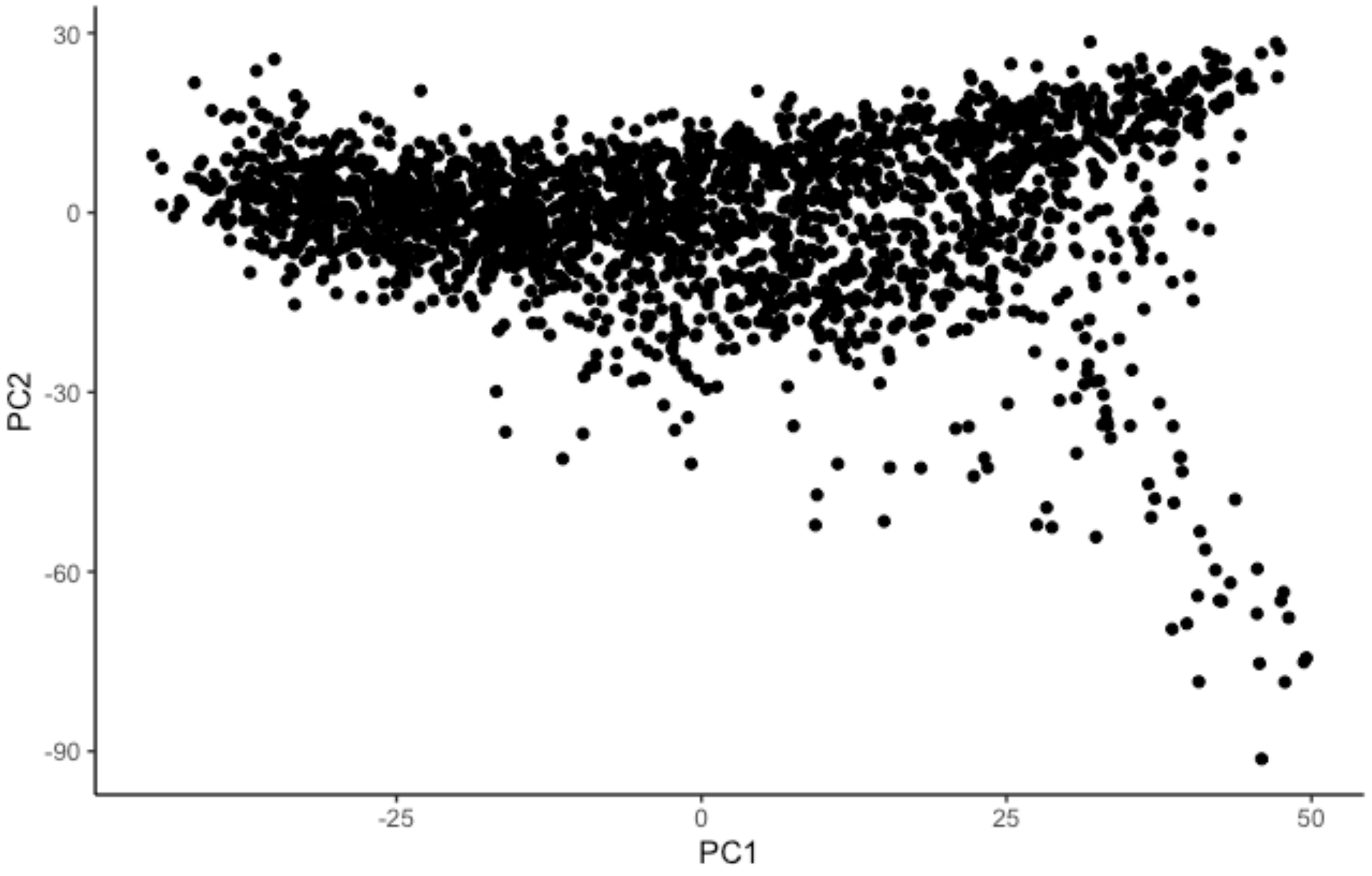
  - Y_1 and Y_2 are *independent (uncorrelated)* and
  - Y_2 captures most of the variability of the data that wasn't already captures by Y_1.
  - And so on for all the remaining components.

NYAKAS ET AL. (2013) MOUSE KIDNEY DATA

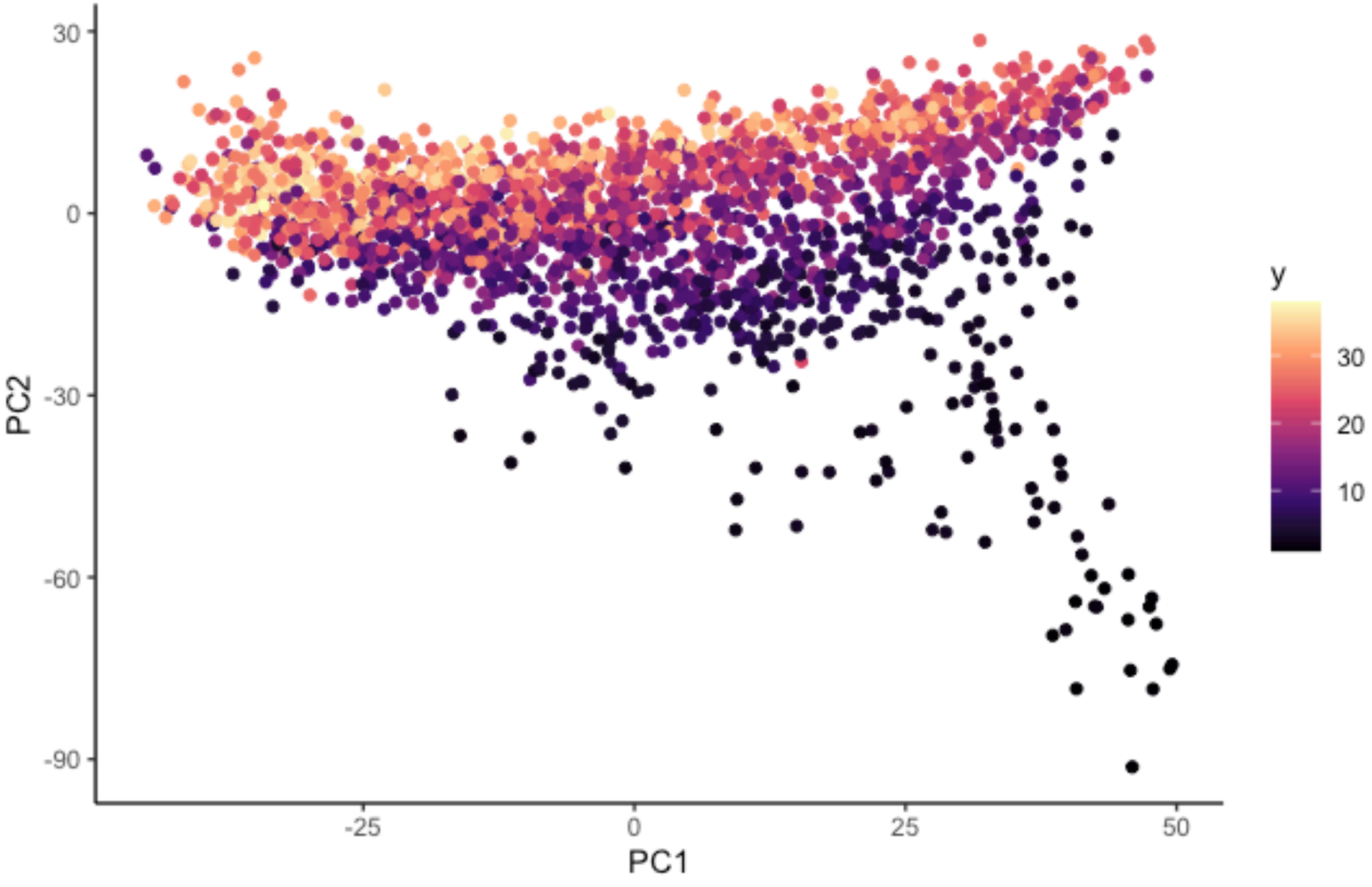


- The data (after preprocessing) is a $(n = 2222) \times (p = 5592)$ matrix.
- Unlike the penguin example, it is impossible to visualize the full data.

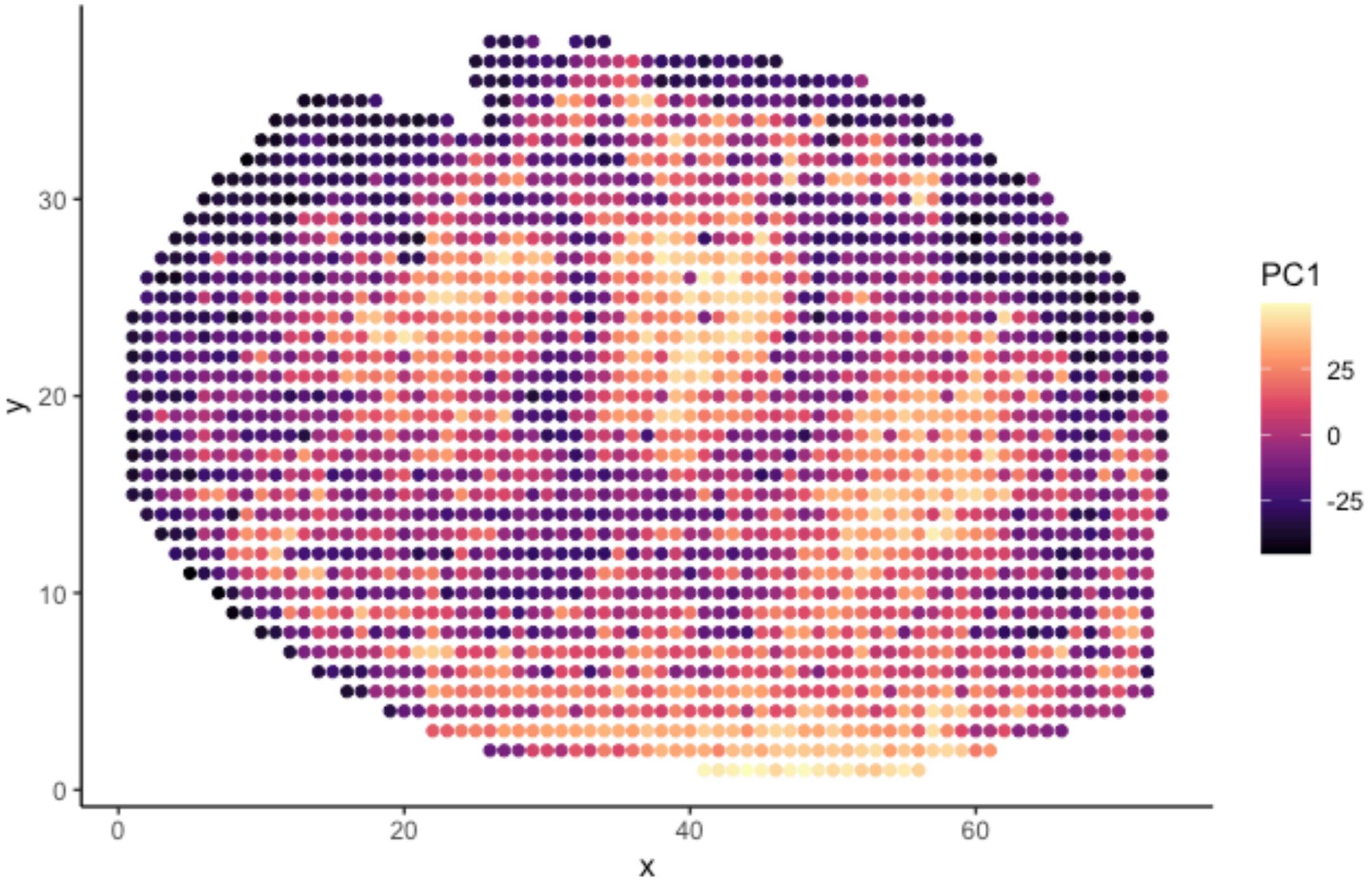
NYAKAS ET AL. (2013) DATA



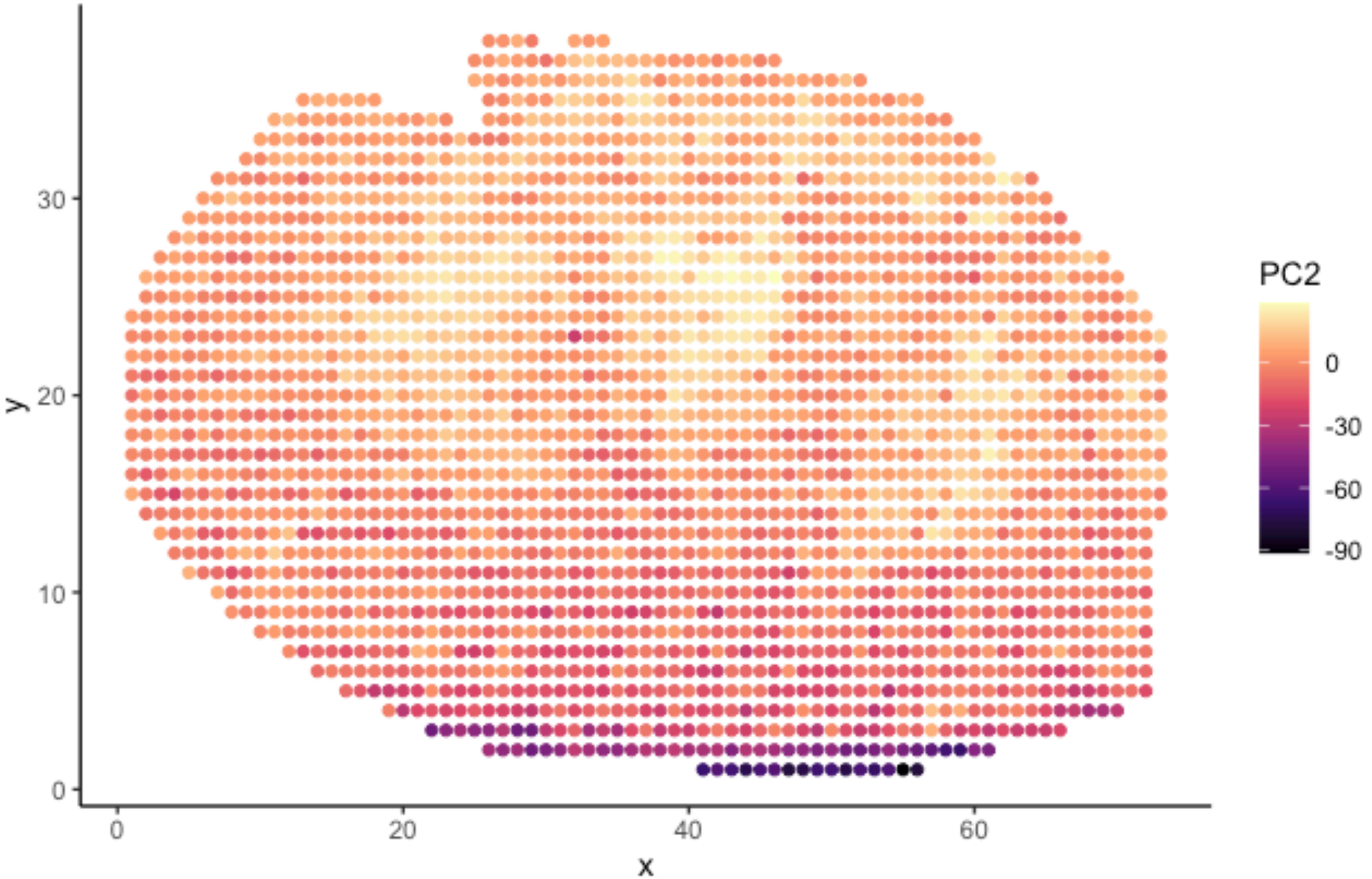
NYAKAS ET AL. (2013) DATA



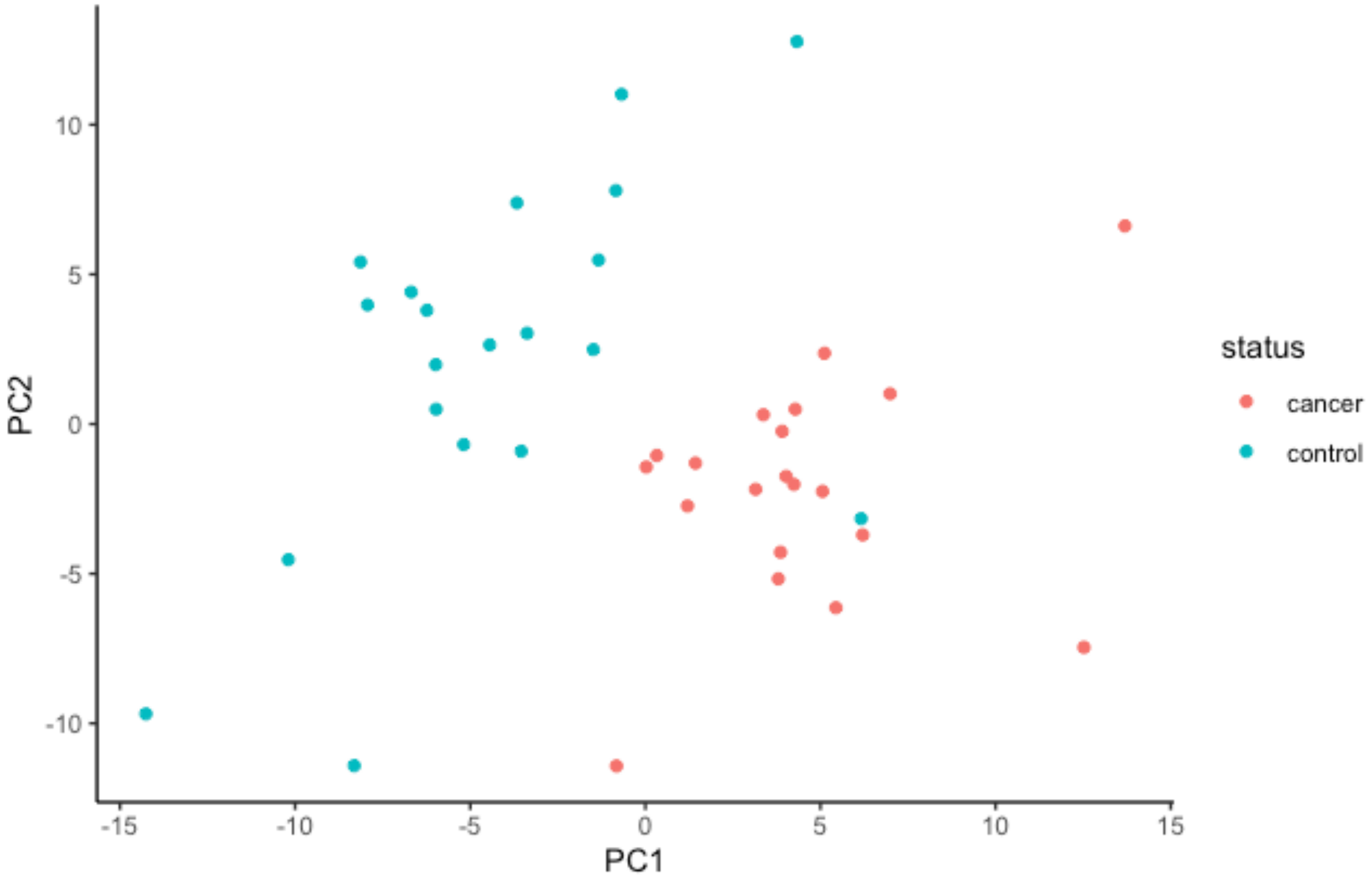
NYAKAS ET AL. (2013) DATA



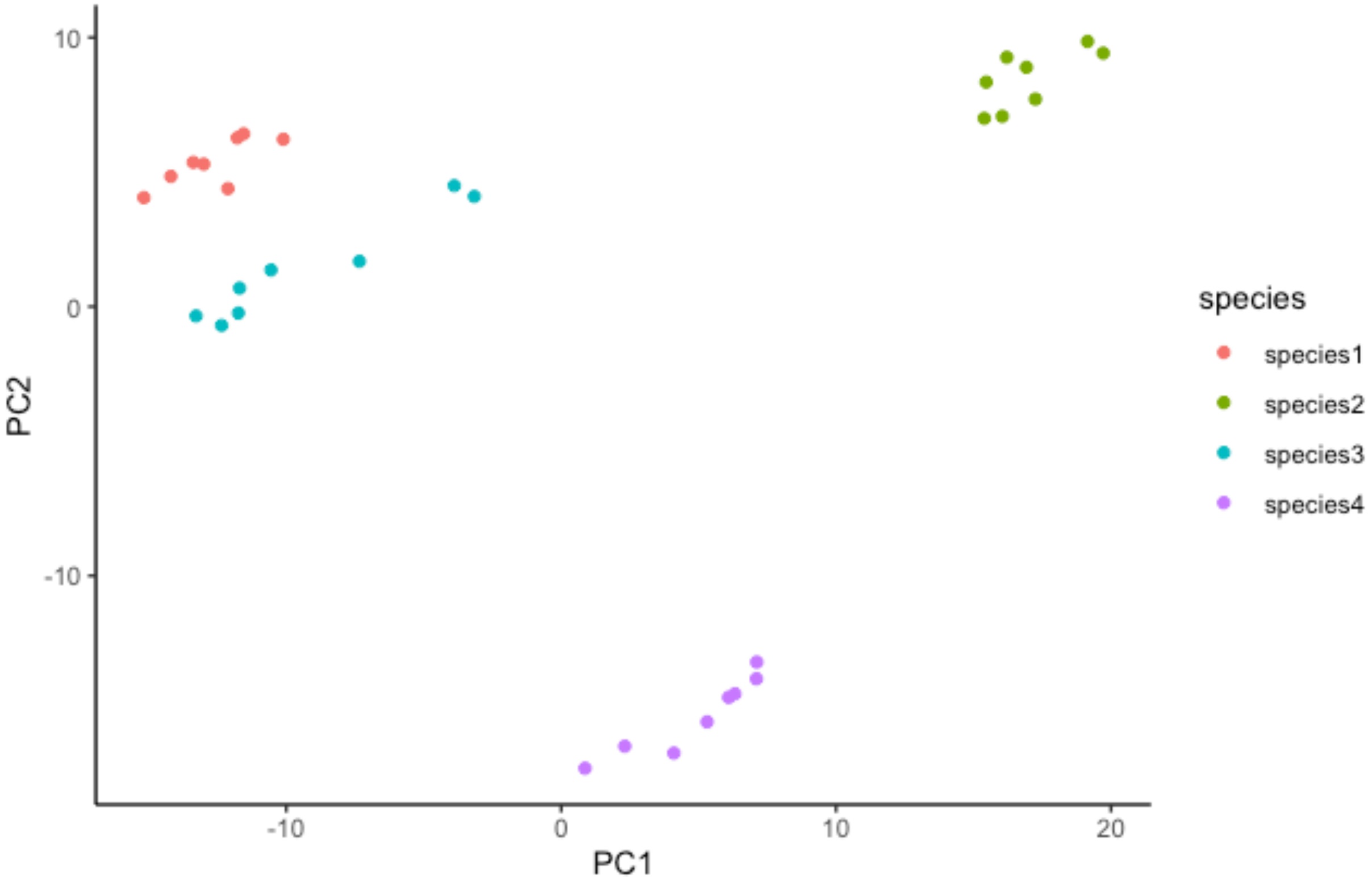
NYAKAS (2013) DATA



FIEDLER ET AL. (2009) DATA



SPECIES DATA



HOW MANY PRINCIPAL COMPONENTS?

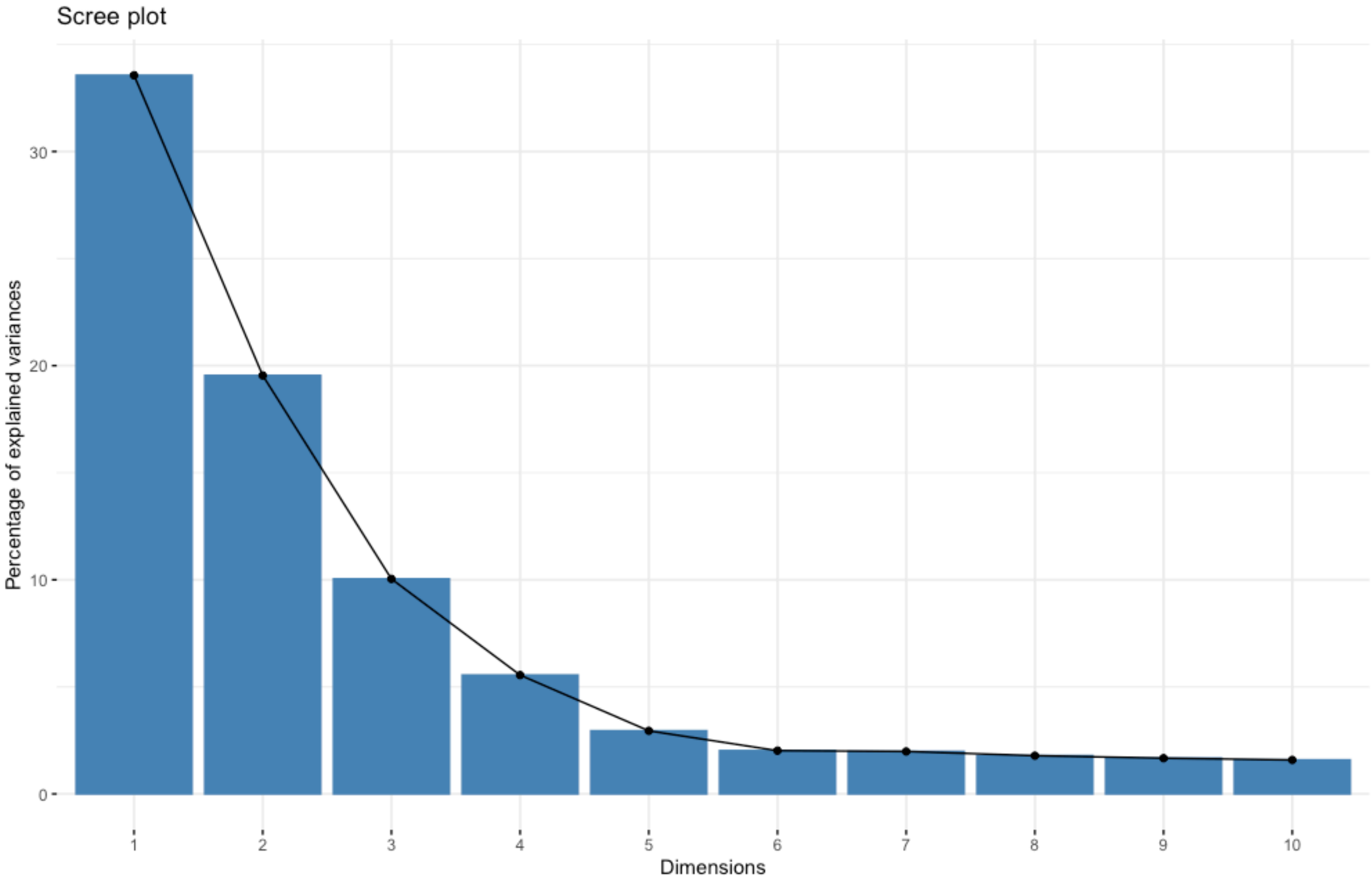
- The number of components to keep in the analysis depends on the goal.
- If we need to visualize the data, it is hard to do so in more than 2-3 dimensions.
- If we want to use the principal components for downstream analyses (e.g., clustering), it may be useful to look at how much variance each of the components explains.

SPECIES DATA

VARIANCE EXPLAINED

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8
Std Dev.	12.26	9.36	6.71	4.99	3.63	3.01	2.98	2.83
Prop. Var.	0.34	0.20	0.10	0.06	0.03	0.02	0.02	0.02
Cum. Prop.	0.34	0.53	0.63	0.69	0.72	0.74	0.76	0.77

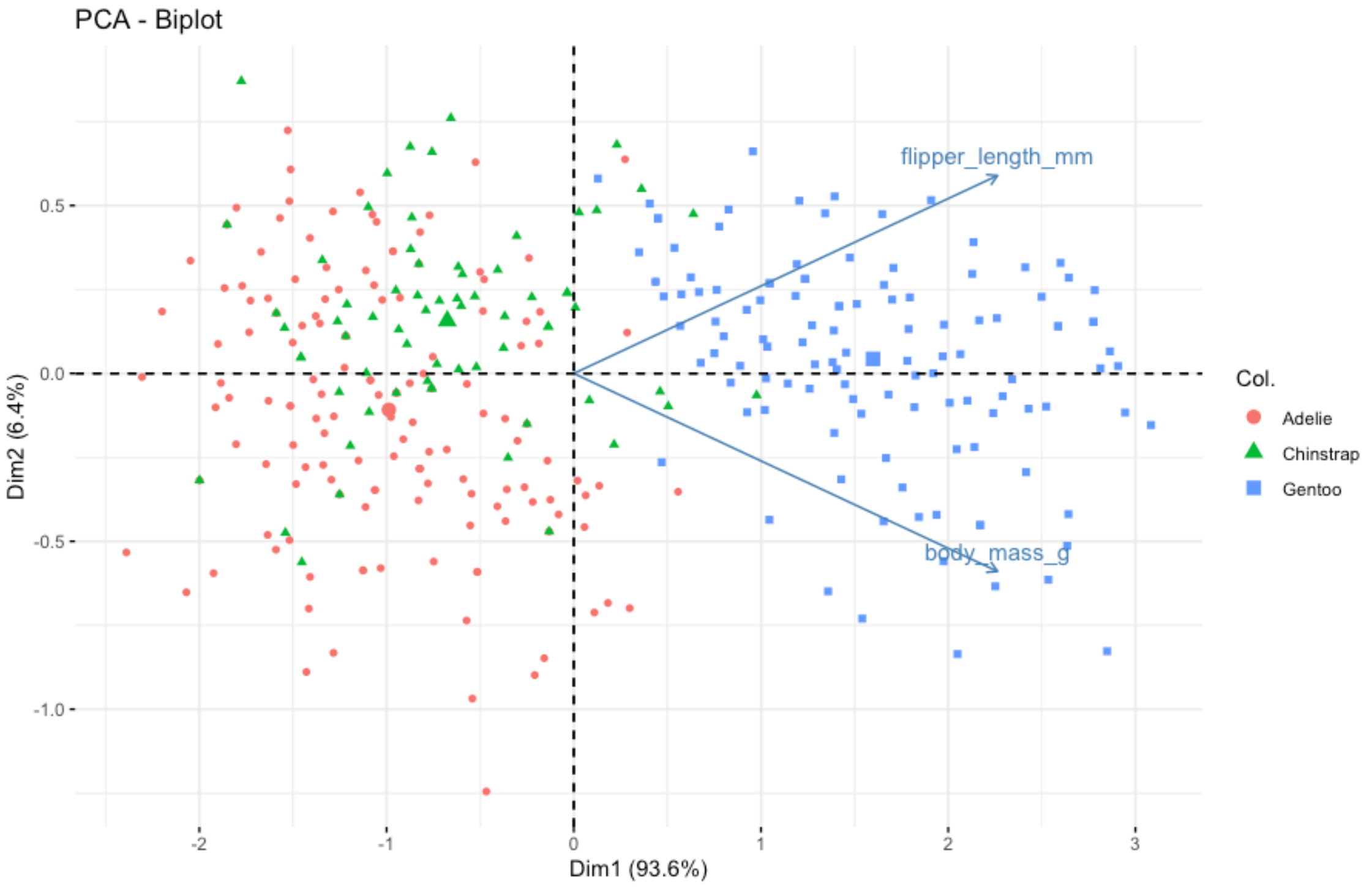
SPECIES DATA



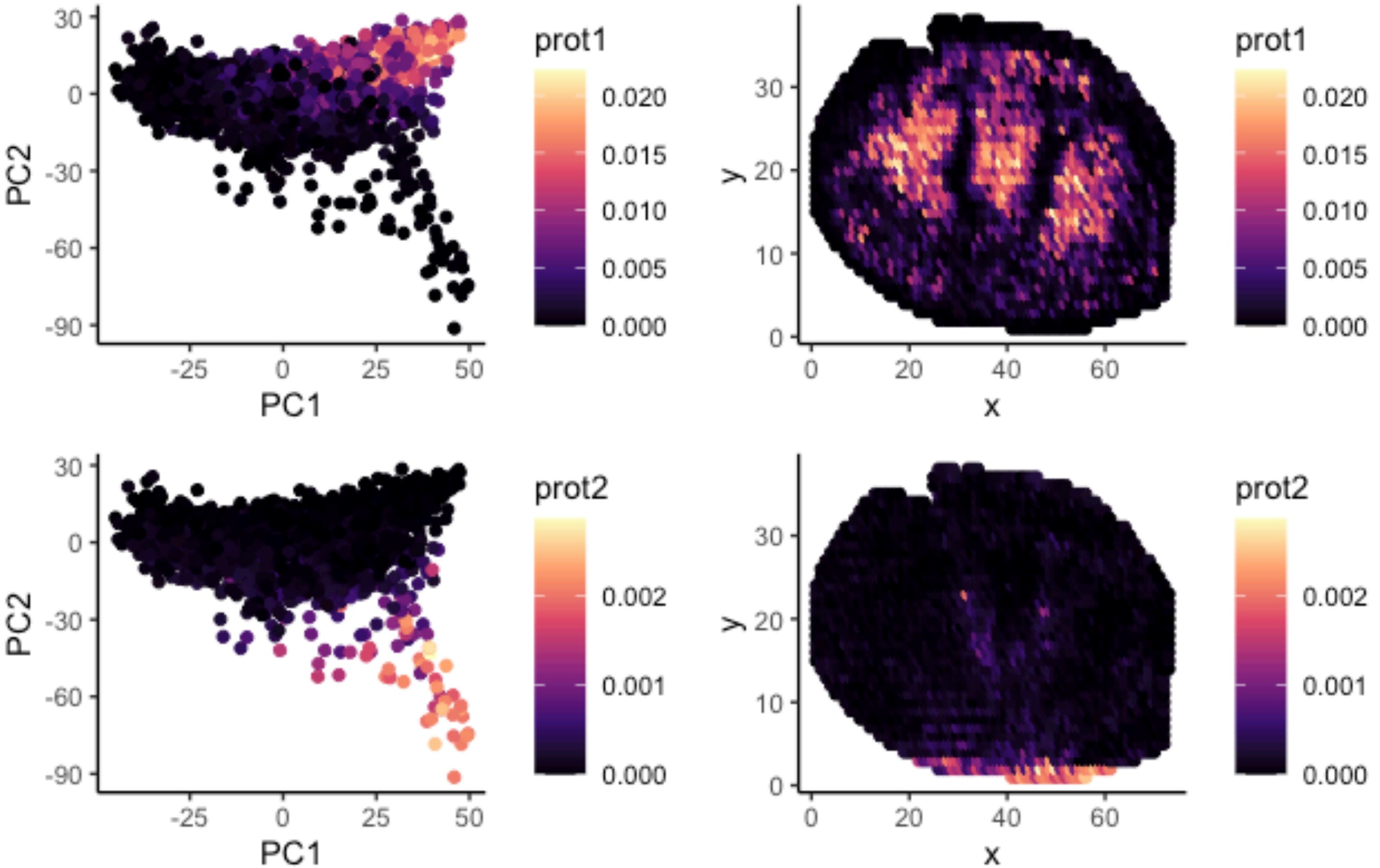
INTERPRETATION OF PRINCIPAL COMPONENTS

- Remember that the *loadings* are the weights of the linear combinations that create the principal components.
- By looking at the sign and magnitude of the loadings we can identify those proteins that most influence each PC.

INTERPRETATION OF PRINCIPAL COMPONENTS – BIPLOT



INTERPRETATION OF PRINCIPAL COMPONENTS



PROS AND CONS OF PCA

- The main feature of PCA is that it is a linear technique.
- This is both its main advantage and drawback:
  - It is advantageous because linearity makes the results interpretable.
  - It is disadvantageous because linear combinations may fail to capture complex patterns in the data.

T-DISTRIBUTED STOCHASTIC NEIGHBOUR EMBEDDING (TSNE)

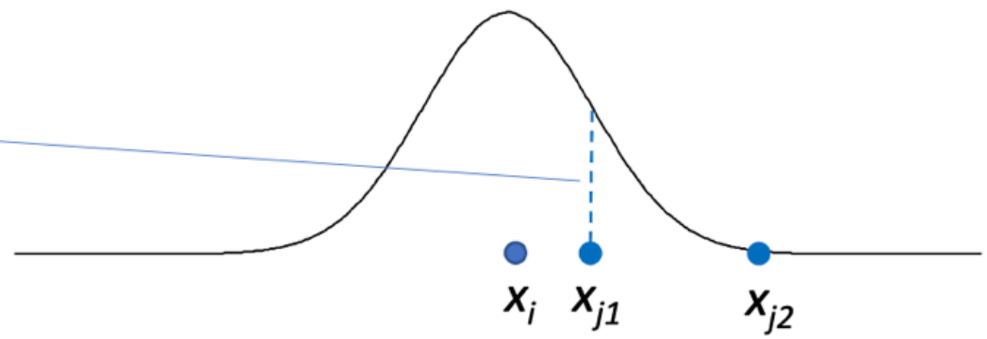
- t-SNE is a non-linear dimensionality reduction technique.
- Briefly, the problem that we want to solve is to represent in a 2-3 dimensional map (embedding) the observations from a high-dimensional space preserving as much as possible the distance between points.

T-DISTRIBUTED STOCHASTIC NEIGHBOUR EMBEDDING (TSNE)

- More formally, the similarity between two points, x_i and x_j, in the original high-dimensional space is defined as the probability that x_i would pick x_j as its neighbour.
- We use a Gaussian kernel.

$$p_{j|i} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\|x_i - x_k\|^2 / 2\sigma_i^2\right)},$$

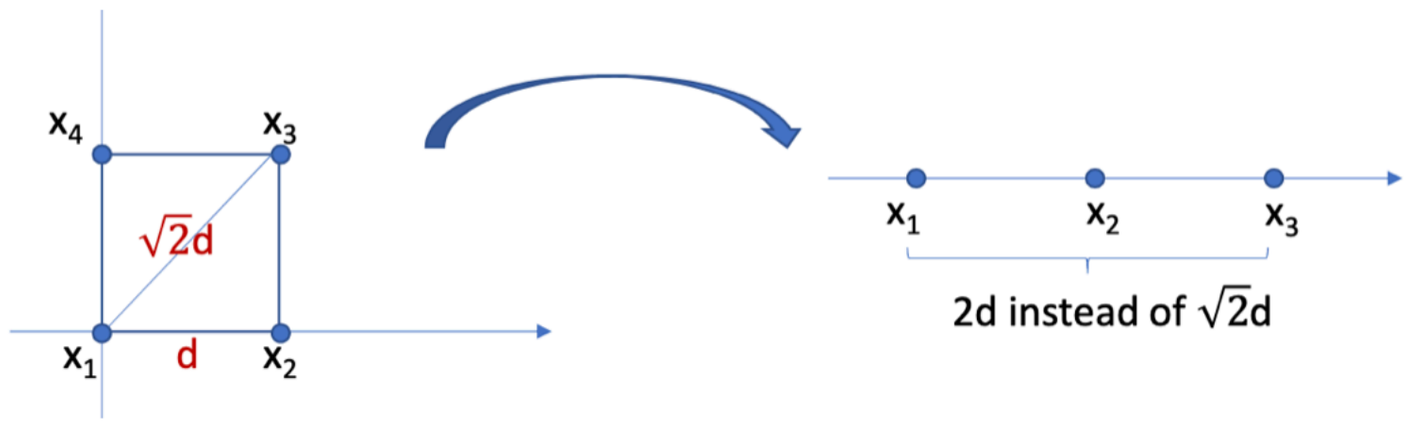
The denominator scales the sum of all the scores to 1



T-DISTRIBUTED STOCHASTIC NEIGHBOUR EMBEDDING (TSNE)

- We could define a similar density in the low-dimensional space, but we use the *t* distribution instead of a Gaussian kernel.
- The *t* distribution has heavier tails and partially accounts for the crowding problem.

$$q_{ij} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k \neq l} \exp\left(-\|y_k - y_l\|^2\right)}, \qquad q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \|y_k - y_l\|^2\right)^{-1}}.$$

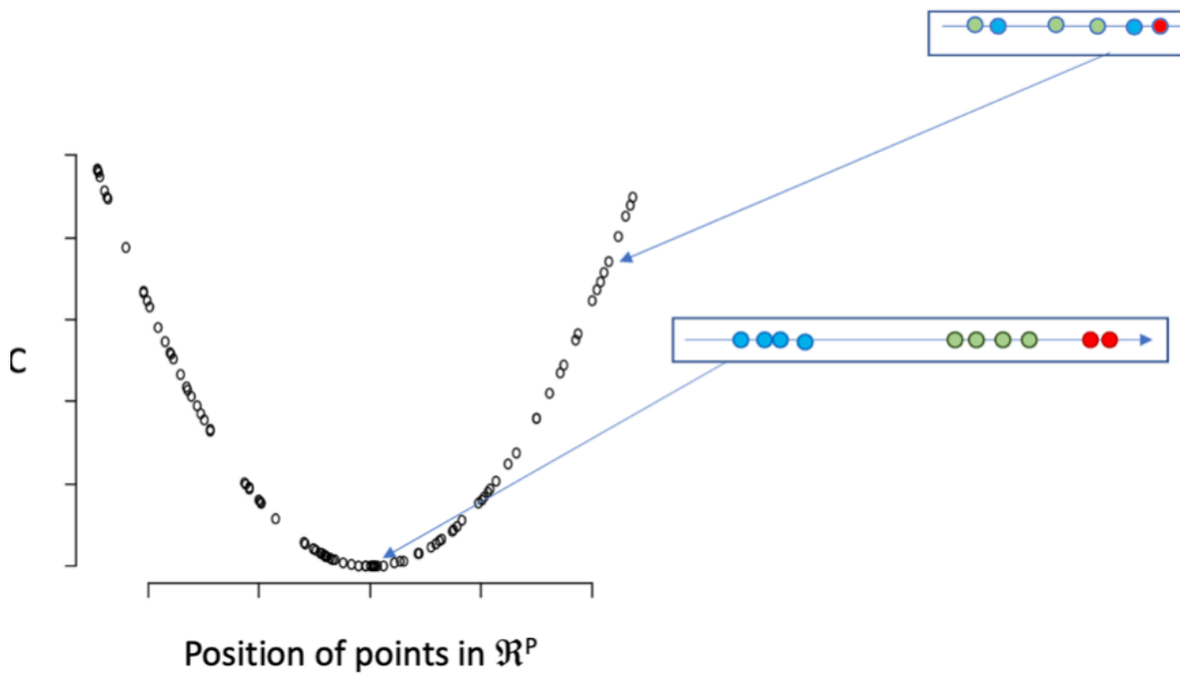


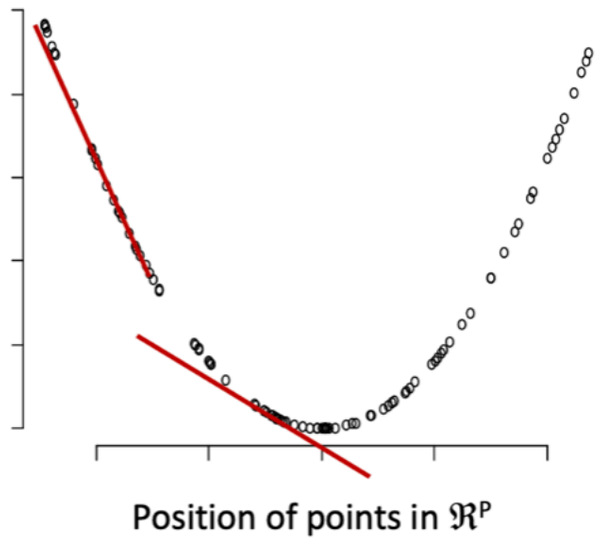
T-DISTRIBUTED STOCHASTIC NEIGHBOUR EMBEDDING (TSNE)

- The algorithm minimizes the Kullblack-Leibler (KL) divergence between the two distributions, using a gradient descent algorithm.

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}.$$

$$q_{ij} = \frac{\exp\left(-\|y_i - y_j\|^2\right)}{\sum_{k \neq l} \exp\left(-\|y_k - y_l\|^2\right)}, \qquad p_{ij} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\sigma^2\right)}{\sum_{k \neq l} \exp\left(-\|x_k - x_l\|^2 / 2\sigma^2\right)},$$

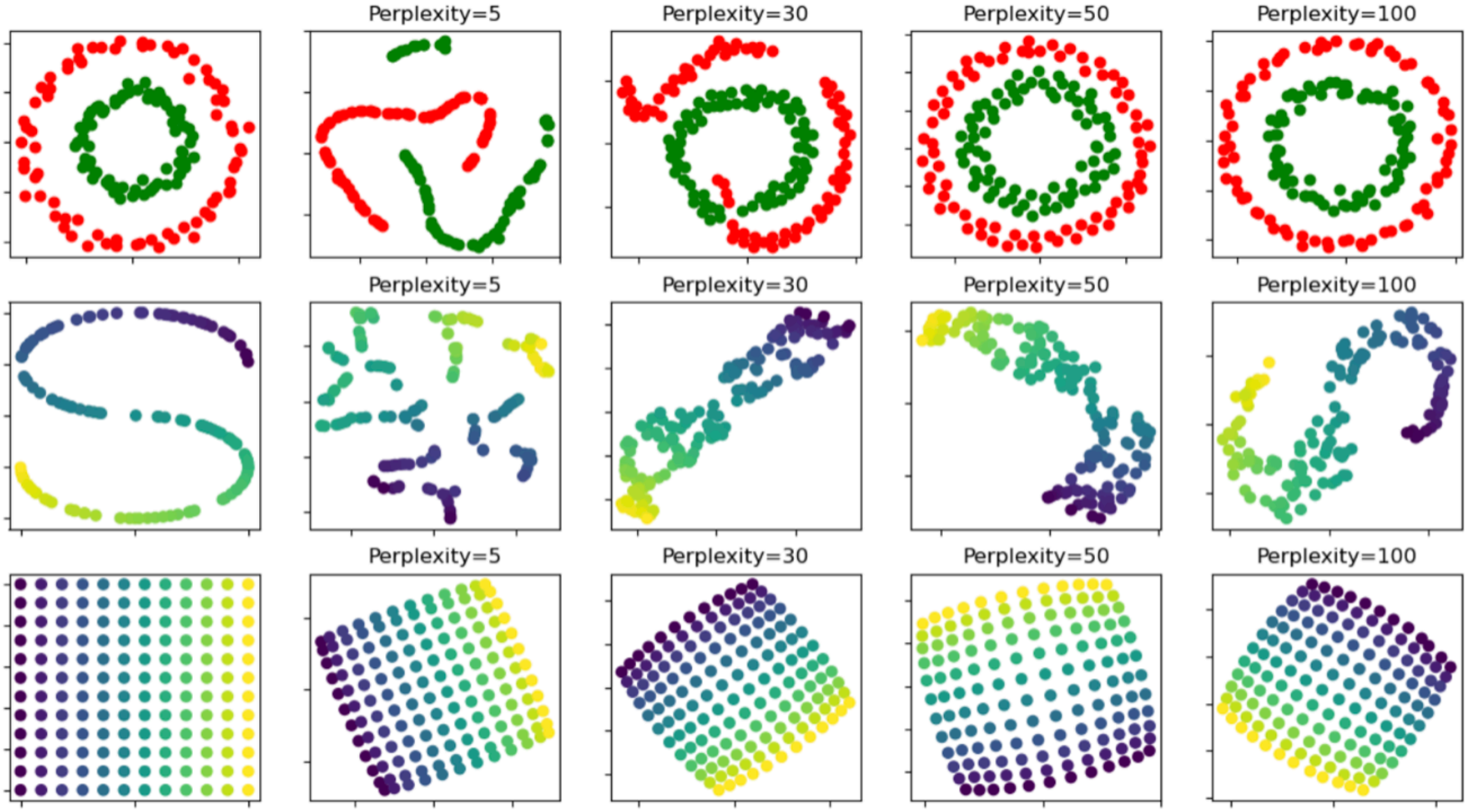




CHOICE OF SCALE PARAMETER

- It’s not appropriate to have a single value for the scale of the kernel as we need smaller values in more dense regions.
- The user can control it through a parameter called perplexity.
- **Perplexity can have a big impact on the results!**

CHOICE OF THE PERPLEXITY PARAMETER



https://distill.pub/2016/misread-tsne

UMAP: UNIFORM MANIFOLD APPROXIMATION AND PROJECTION FOR DIMENSION REDUCTION

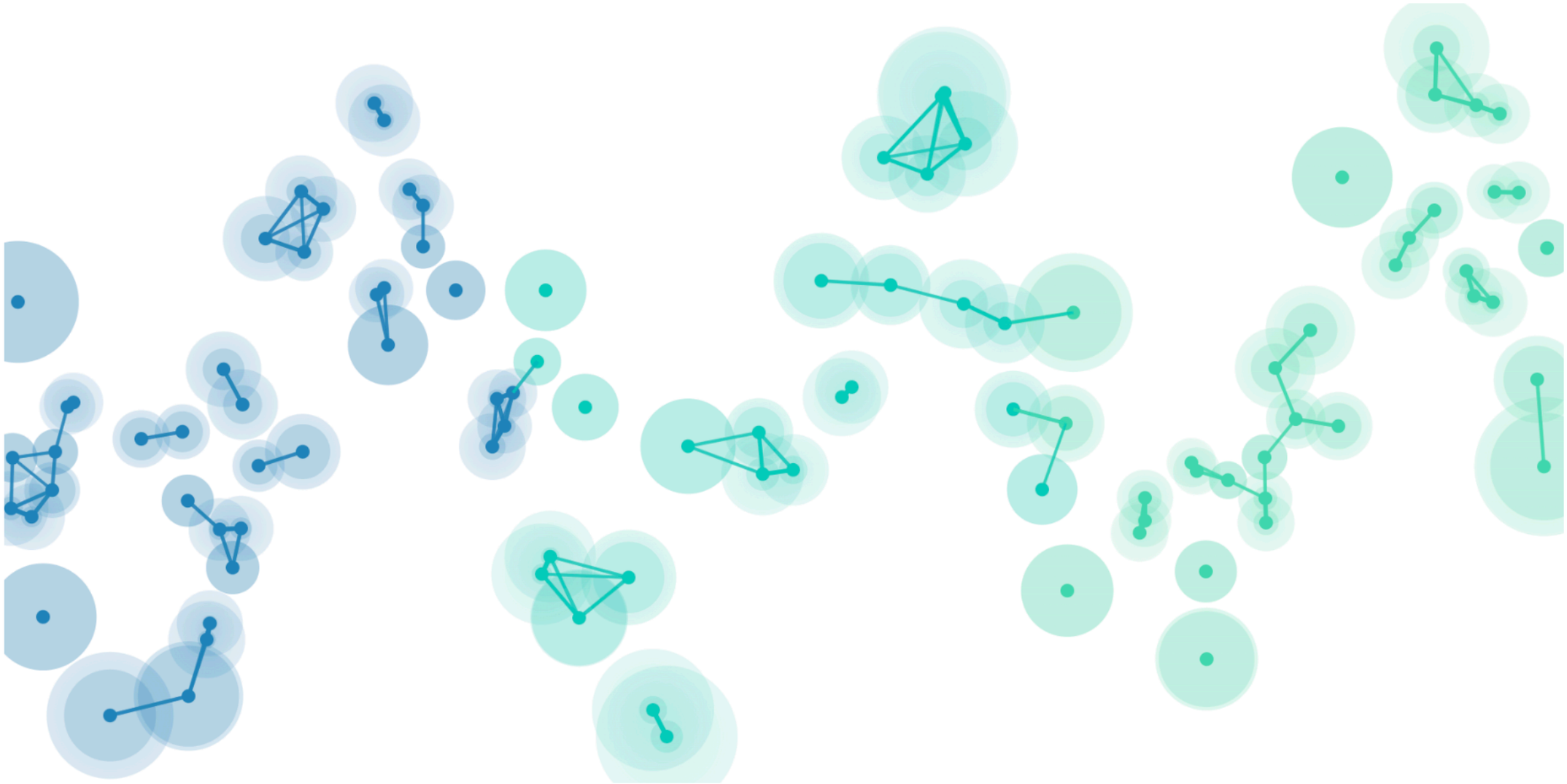
- UMAP is a technique similar to t-SNE that has gained popularity thanks to its scalability and claims that it preserves the global structure of the data better than t-SNE.
- It uses a similar algorithm but constructs a low-dimensional graph to be as structurally similar as possible to a high-dimensional graph representation of the data.
- UMAP uses a “fuzzy simplicial complex” instead of the Gaussian kernel to create the high-dimensional distance graph.

https://pair-code.github.io/understanding-umap/

UMAP: UNIFORM MANIFOLD APPROXIMATION AND PROJECTION FOR DIMENSION REDUCTION

- The idea is to create a circle centered at each point and connect the points for which the circles overlap.
- By increasing the radius of the circles, more and more points will connect.
- The graph is made “fuzzy” by decreasing the likelihood of connection as the radius grows.
- By ensuring that each point must be connected to at least its closest neighbor, UMAP ensures that local structure is preserved in balance with global structure.

UMAP: UNIFORM MANIFOLD APPROXIMATION AND PROJECTION FOR DIMENSION REDUCTION



extent: 33%

n_nearest: 5

UMAP PARAMETERS

- UMAP has two parameters that play an important role and influence the results.
- **The number of neighbors** to use to construct the initial high-dimensional graph.
  - Low values will push UMAP to focus more on local structure, and high values on global structure
- **The minimum distance** between points in low-dimensional space.
  - Smaller values will “squish” the points closer together.

CHOICE OF THE UMAP AND T-SNE PARAMETERS

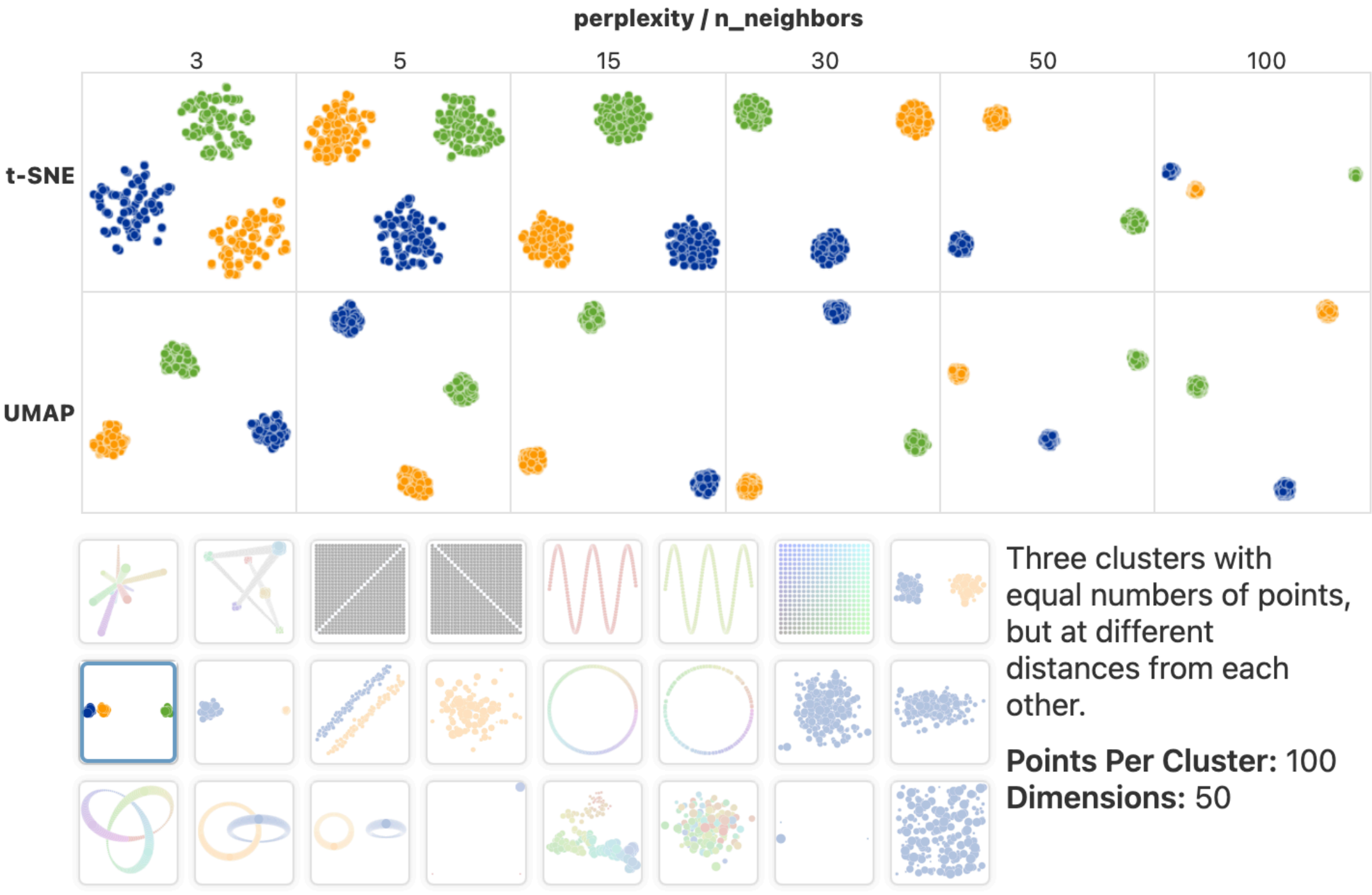


Figure 7: *Comparison between UMAP and t-SNE projecting various toy datasets.*

https://pair-code.github.io/understanding-umap/

LIMITATIONS OF T-SNE AND UMAP

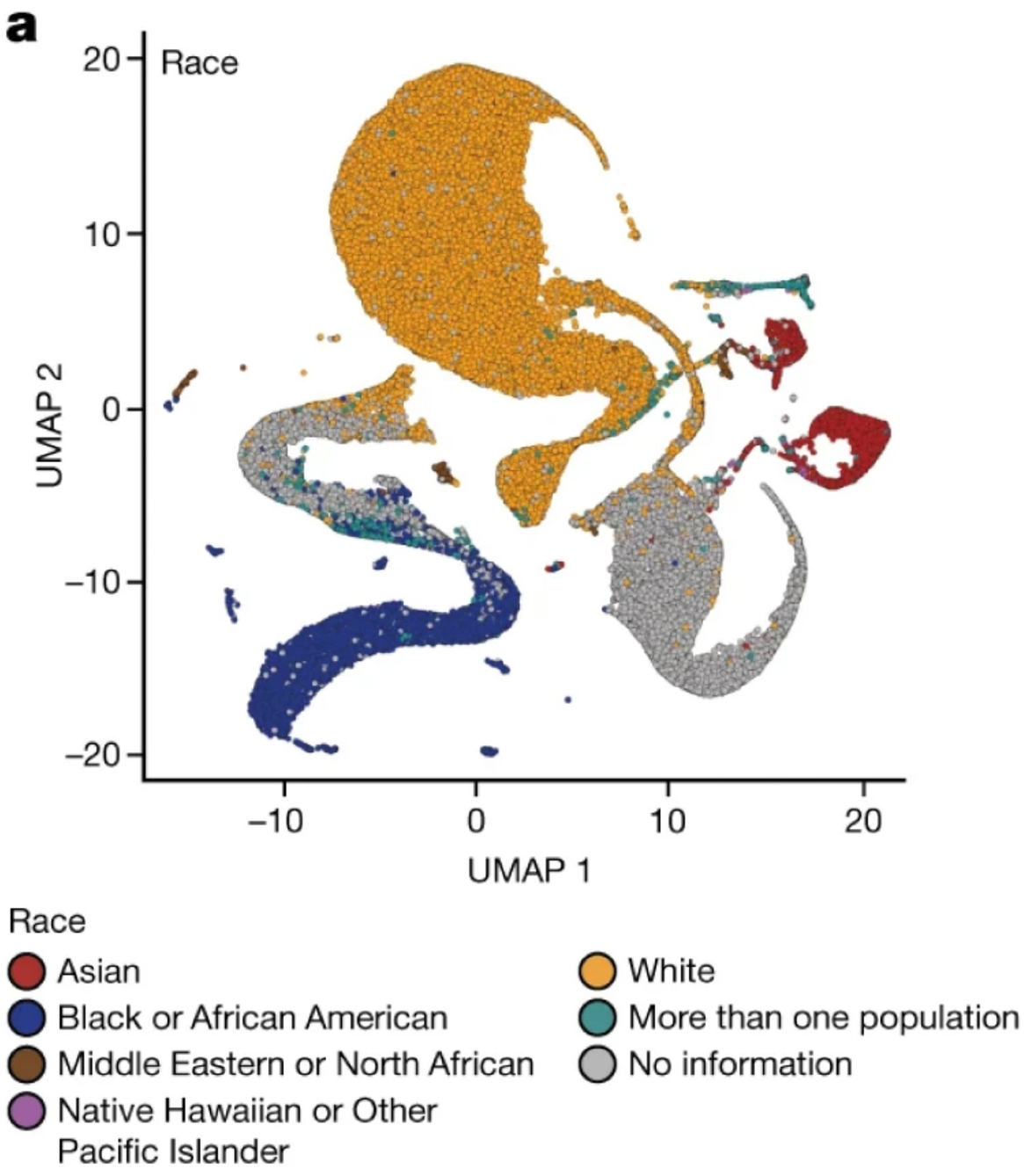
- Unlike PCA, we do not have a simple interpretation for the low-dimensional embedding (the axes have "no meaning").
- t-SNE preserves only the local structure (who is the neighbor of whom) but not the global structure.
- There is no guarantee of convergence to a global minimum (non-convex problem), hence the different runs will lead to different embeddings.
- The "shape" of the data in the embedding is arbitrary.

HOW TO MISREAD UMAP (AND T-SNE)

- Parameters can make a huge impact on the results!
- Cluster sizes in the plot mean nothing!
- Distances between clusters mean (almost) nothing!
- Random noise doesn't always look random! (e.g., it may form additional "interesting" clusters)

- See also: https://doi.org/10.1371/journal.pcbi.1011288

HOW TO MISREAD UMAP (AND T-SNE)

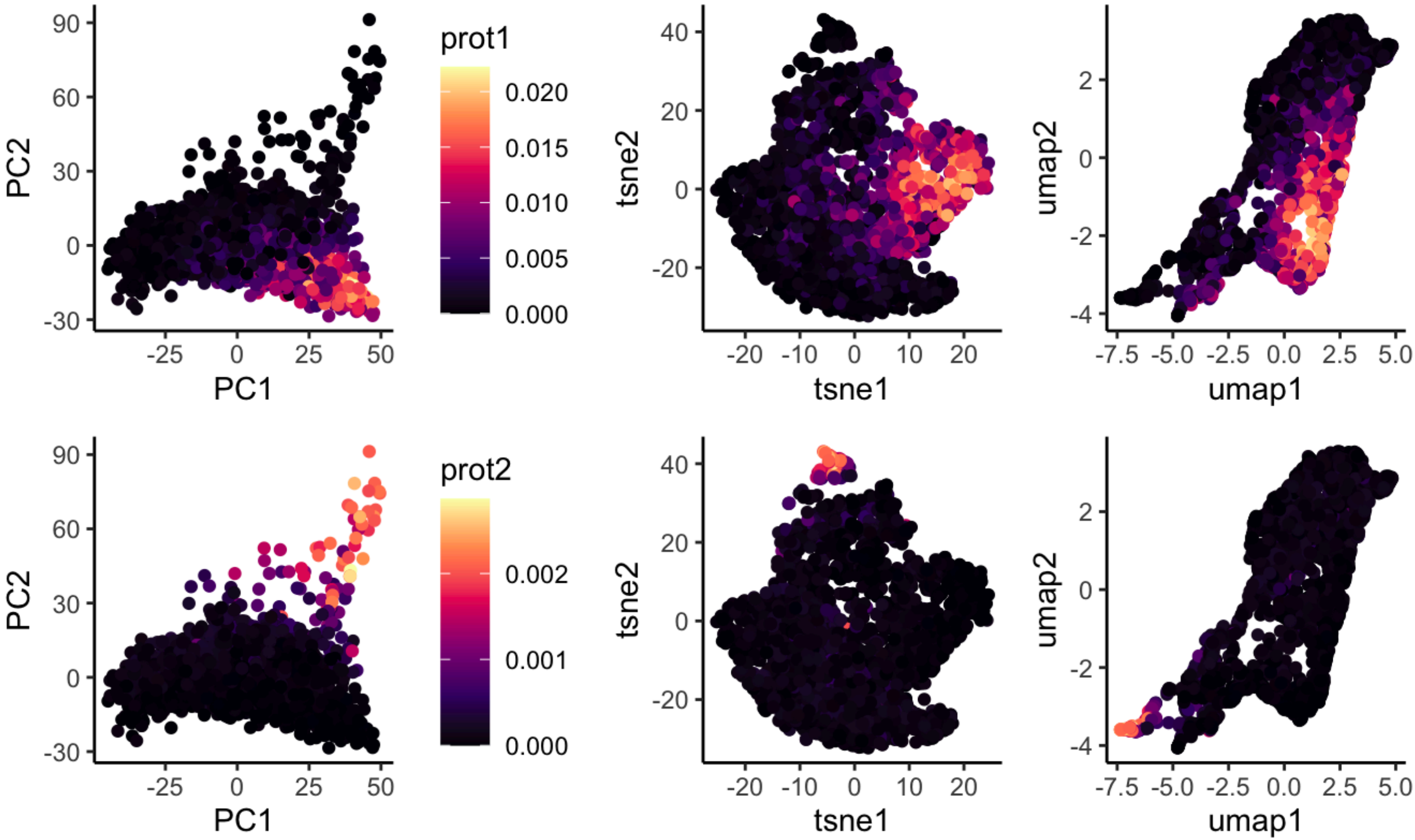


UMAP of 250,000 people in the “All of Us” study.

The paper’s corresponding author [...] points out that the three other major human genome papers published in the past few years [...] also use the UMAP algorithm, which “is frankly why we selected it.”

https://www.science.org/content/article/huge-genome-study-confronted-concerns-over-race-analysis

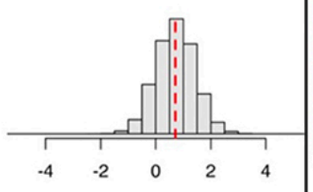
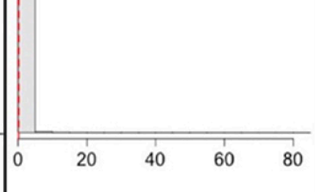
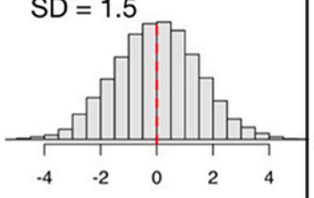
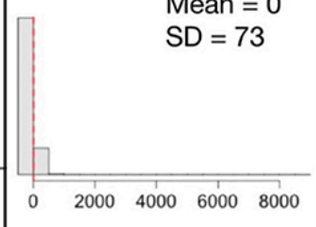
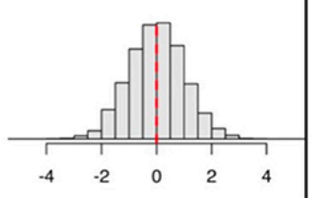
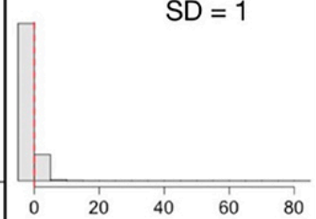
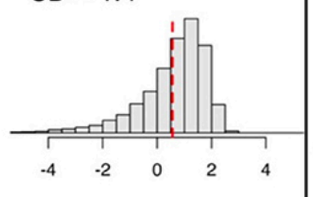
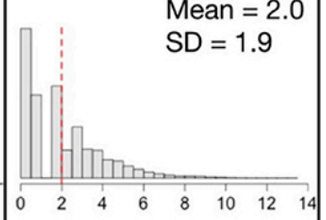
NYAKAS (2013) DATA



TO SCALE OR NOT TO SCALE

- PCA corresponds to the *spectral decomposition* of the covariance (or correlation) matrix.
- This corresponds to the PCA of the data after *centering* and potentially *scaling* the data.
- Scaling (or standardizing) means transforming the data so that the variables have unit variance.
- Whether you scale or not is a question that depends on the application:
  - *By scaling we focus on correlation rather than covariance.*
  - *Is the magnitude of the variation something to focus on or to normalize against?*

WHAT ARE WE ACTUALLY DOING WHEN WE SCALE?

Method	Formula	Example raw datasets Graphical Examples Toy data	scMix, 10X counts
Scale	$x_{i,j}{}^{*} = \frac{x_{i,j}}{scaling\ factor}$ where the *scaling factor* can be a size or data dispersion measure	Mean = 0.7 SD = 0.7 	Mean = 0.2 SD = 1.0 
Center	$x_{i,j}{}^{*} = x_{i,j} - column\ mean$	Mean = 0 SD = 1.5 	Mean = 0 SD = 73 
Standardize	$x_{i,j}{}^{*} = \frac{x_{i,j} - column\ mean}{scaling\ factor}$ where the *scaling factor* can be a size or data dispersion measure. For example z-score subtracts means, divides by standard deviation	Mean = 0 SD = 1 	Mean = 0 SD = 1 
Transform	$x_{i,j}{}^{*} = f(x_{i,j})$ where *f(x)* is the transformation function, for example logarithms are commonly used	Mean = 0.5 SD = 1.4  *Log_2 transformation*	Mean = 2.0 SD = 1.9  *Log_2 transformation, pseudocount of 1*

[Hsu and Culhane (2020)]

WHAT TO DO WITH COVARIATES?

- Often, proteomics data are not the only variables that we have about our samples. For instance, we could have genotype, clinical characteristics, or other information.
- It may be important to take into account such variables in the dimensionality reduction.
- The most important thing is to think about the biological question that we are trying to answer and the design of our experiment.
- If our additional variables represent a signal that we want to include in the low-dimensional representation, we can include them in the data matrix e.g. using the ***PCAmixdata*** package if they are not continuous.
- If our additional variables represent nuisance factors that we want to account for in the representation, we can add them as covariates, e.g., using the ***glmPCA*** and ***gllvm*** packages.

CLUSTERING

INTRODUCTION

- Research question: is the population from which the data are measured homogeneous? Or can we identify a number of subpopulations with specific characteristics?
- Cluster analysis (or clustering) tries to group the *n* statistical units into *k* homogeneous clusters (groups).
- We want the groups to be homogeneous and distinct:
  - "Similar" observations should belong to the same group;
  - "Different" observations should belong to different groups.

SIMILARITY: DISTANCE?

- We need to clarify what we mean by “similar” and “different”.
- We need to define a measure of similarity, or better yet, a distance between two observations.
- This is hard in proteomics:
  - Unclear how to define a distance;
  - The algorithms are computationally complex;
  - The space in which to compute the distance is huge.

WHAT IS A DISTANCE?

- Let's assume that we have two points x and y (i.e., two samples).
- A real-valued function $d(x, y)$ is a distance if the following properties hold:

1. Symmetry: $d(x, y) = d(y, x)$;
2. Non-negativity: $d(x, y) \geq 0$;
3. Identity: $d(x, x) = 0$;
4. Triangular inequality: $d(x, y) \leq d(x, z) + d(z, y)$

WHICH DISTANCE?

- The most used distance is the Euclidean distance:

$$d(x, y) = \sqrt{\sum_{i=1}^{p} (x_i - y_i)^2}$$

- There are many other possible choices, e.g., the Manhattan distance, and the cosine distance (1 - correlation).

WHICH SPACE?

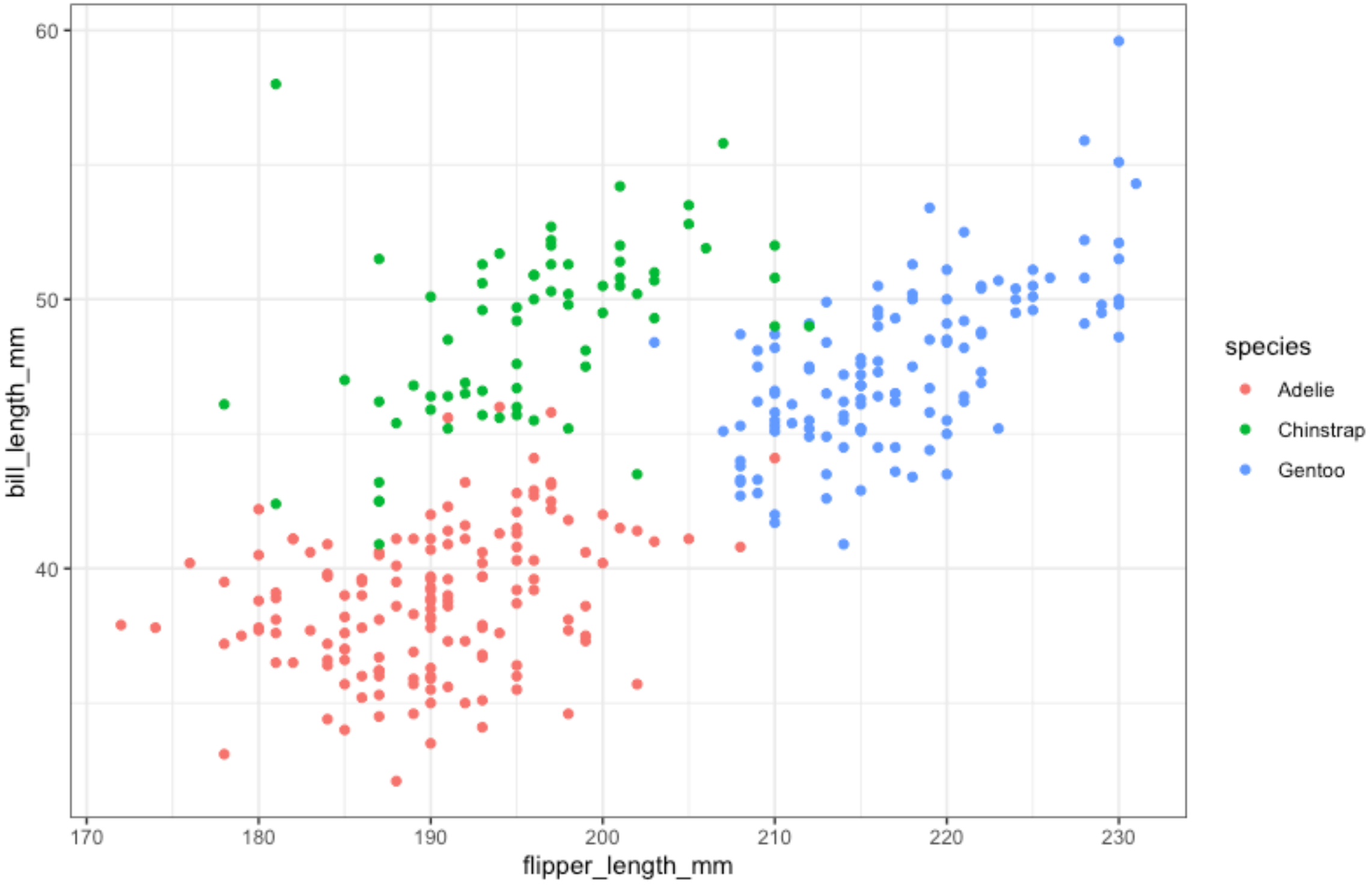
- Remember that we are measuring thousands of proteins, hence the p in the sum in the previous slide is very large.
- This is problematic if we think about it in geometric sense: if we measure 5,000 proteins, we are measuring distances in a 5,000-dimensional space!
- When we deal with such huge spaces, we observe a phenomenon known as *the curse of dimensionality*.

THE CURSE OF DIMENSIONALITY

There are several ways to define this “curse”, but with reference to the Euclidean distance, it is intuitive to think that the space becomes so vast that distances become meaningless, as “no-one is close to anyone”.

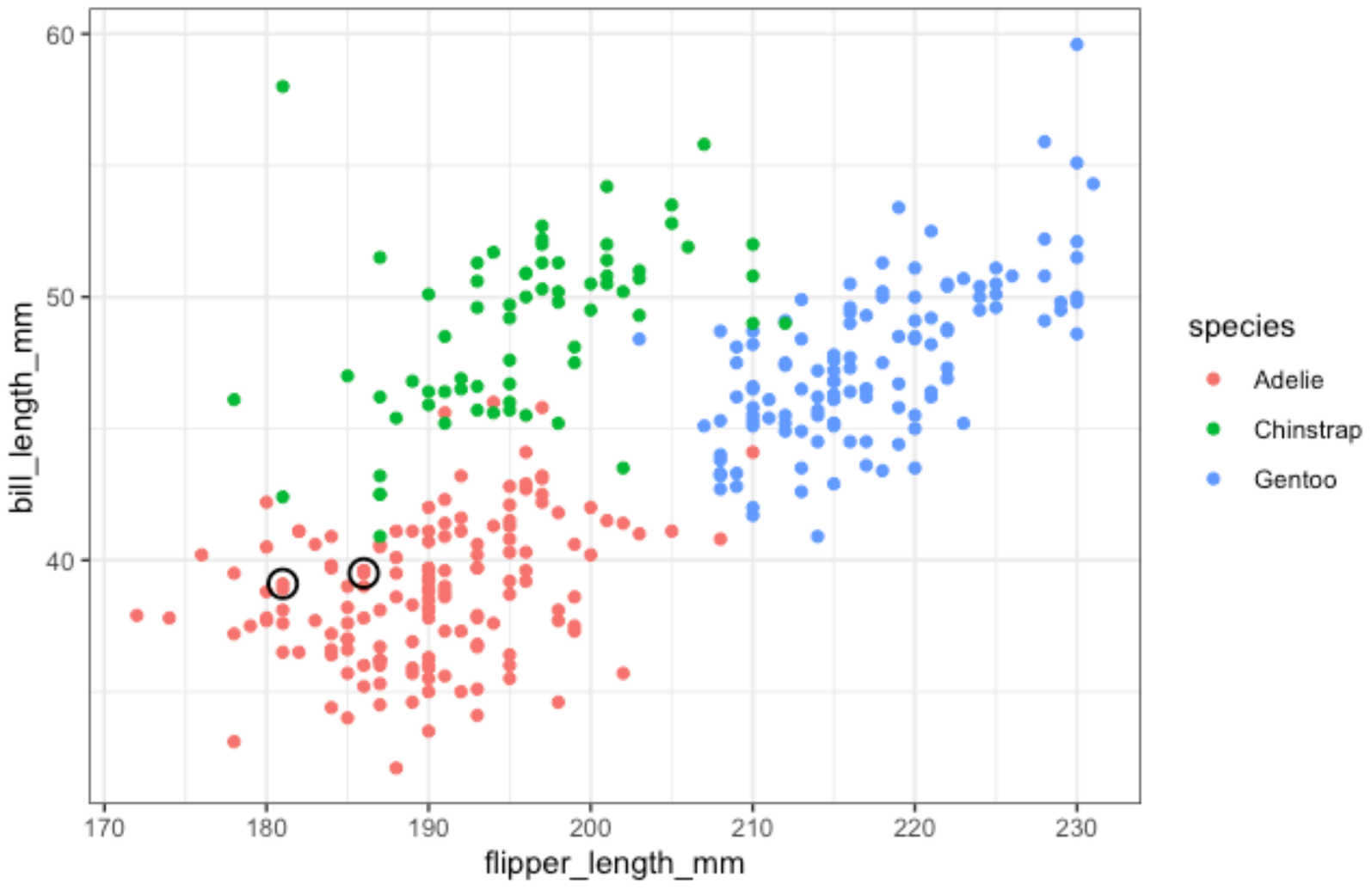
For this reason, clustering is usually applied after an initial **dimensionality reduction step**.

BACK TO PENGUINS



BACK TO PENGUINS

- We know how to compute distances between two penguins, e.g.,



$$d(x, y) = \sqrt{(181 - 186)^2 + (39.1 - 39.5)^2} = 5.02$$

DISTANCE BETWEEN TWO GROUPS OF POINTS

- There are several options, the most used are:
- **Single linkage:** the distance between the two closest points;
- **Complete linkage:** the distance between the two points which are farthest away;
- **Average linkage (Ward):** the average of the distances between all possible pairs of points. (Ward's distance also suitably considers the dimension of the groups.)

HIERARCHICAL CLUSTERING

- In the beginning, every unit forms its own group. There are hence n groups.
- The two "closest" statistical units are merged; we now have $n - 1$groups.
- The two "closest" groups (may these be two statistical units or the group of dimension two we just formed at the previous step) are merged; there are now $n - 2$ groups.
- And so on. At every step, the two closest groups are merged.
- We successively have $n - 3, n - 4, \ldots$ groups until all statistical units are "merged" into the same group. (That is, we stop with a single group that includes all units.)
- We just described an "agglomerative" algorithm. We can also proceed top-down, though this is less common. In this case the algorithm is called "divisive".

HIERARCHICAL CLUSTERING: TOY EXAMPLE

- Let's assume we have 5 observations and that we computed all the pairwise distances.

$$\mathbf{D} = \begin{array}{c} \begin{array}{ccccc} O_1 & O_2 & O_3 & O_4 & O_5 \end{array} \\ \begin{bmatrix} 0 & 9 & 3 & 6 & 11 \\ & 0 & 7 & 5 & 10 \\ & & 0 & 9 & 2 \\ & & & 0 & 8 \\ & & & & 0 \end{bmatrix} \end{array} \begin{array}{l} O_1 \\ O_2 \\ O_3 \\ O_4 \\ O_5 \end{array}$$

- O_3 and O_5 are the closest points, hence they get merged.
- We can use the *single linkage* and compute the distance between each observation and the new group as the minimum distance.

HIERARCHICAL CLUSTERING: TOY EXAMPLE

- The new distance matrix is

$$\mathbf{D} = \begin{array}{c} \begin{matrix} O_1 & O_2 & O_4 & (O_3, O_5) \end{matrix} \\ \begin{bmatrix} 0 & 9 & 6 & 3 \\ & 0 & 5 & 7 \\ & & 0 & 8 \\ & & & 0 \end{bmatrix} \end{array} \begin{matrix} O_1 \\ O_2 \\ O_4 \\ (O_3, O_5) \end{matrix}$$

- Now the minimum distance is between O_1 and the group (O_3, O_5) . A new group is formed.

$$\mathbf{D} = \begin{array}{c} \begin{matrix} O_2 & O_4 & (O_1, O_3, O_5) \end{matrix} \\ \begin{bmatrix} 0 & 5 & 7 \\ & 0 & 6 \\ & & 0 \end{bmatrix} \end{array} \begin{matrix} O_2 \\ O_4 \\ (O_1, O_3, O_5) \end{matrix}$$

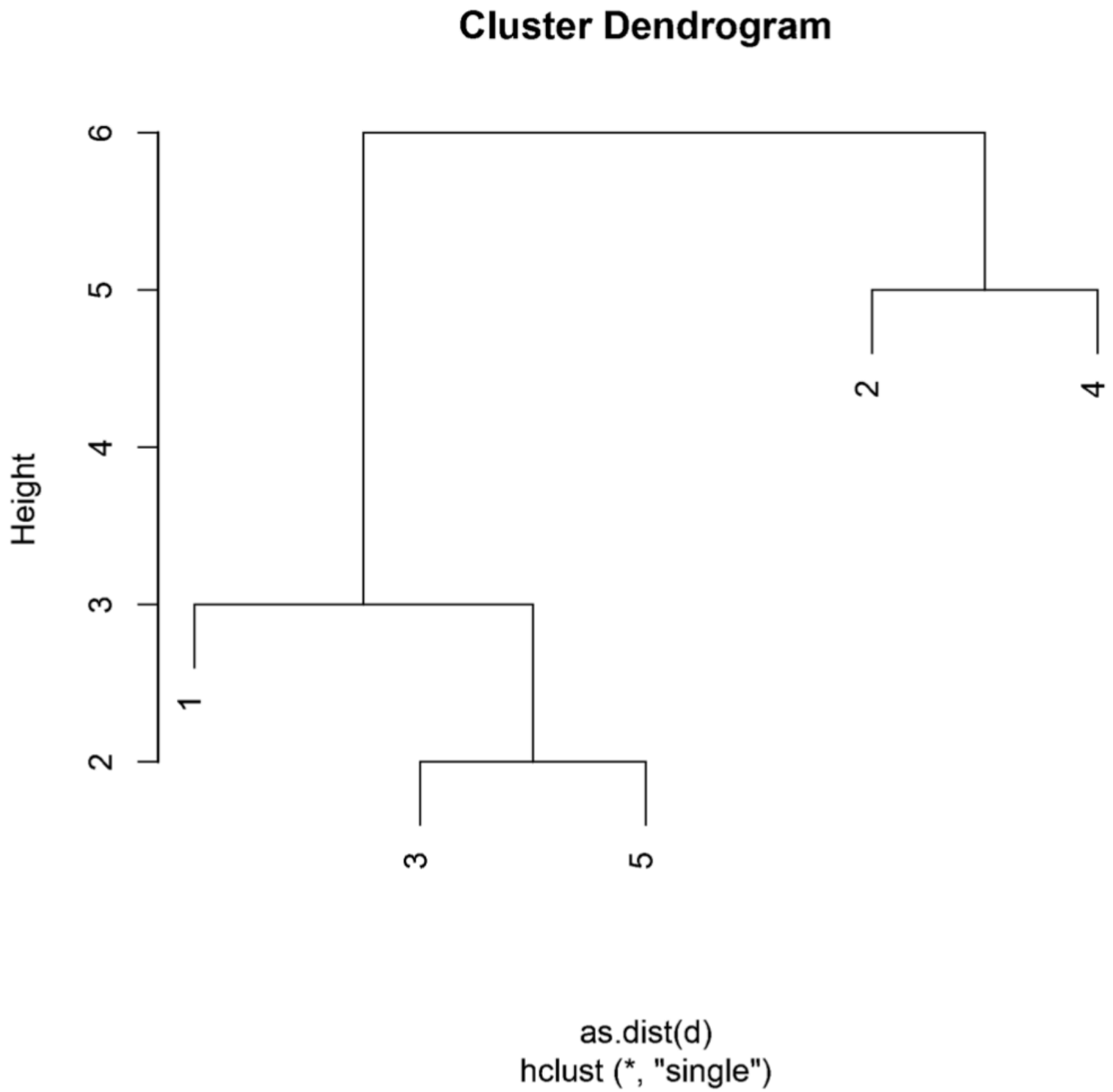
HIERARCHICAL CLUSTERING: TOY EXAMPLE

- Then we merge O_2 and O_4

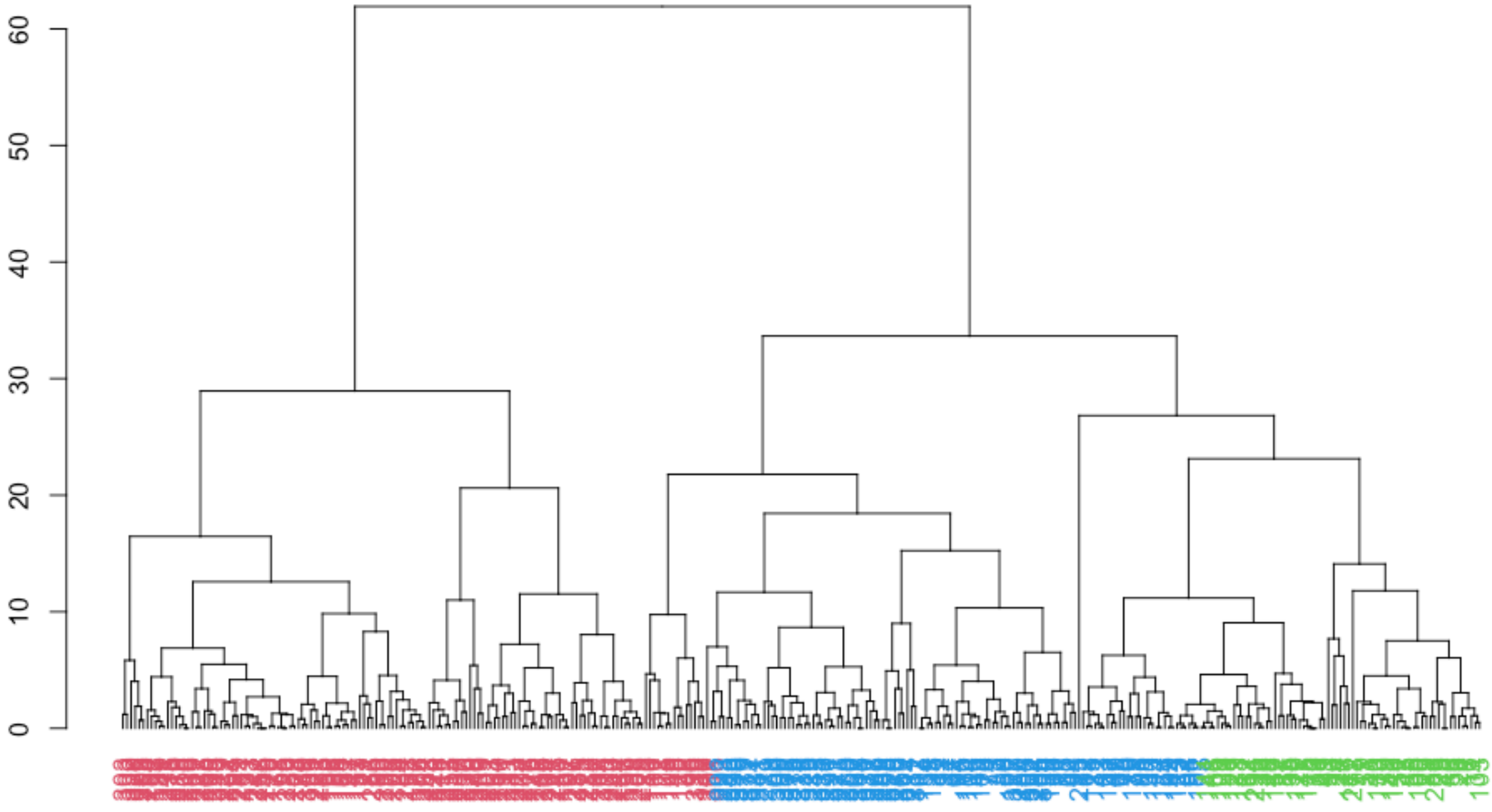
$$\mathbf{D} = \begin{array}{c} \begin{matrix} (O_2, O_4) & (O_1, O_3, O_5) \end{matrix} \\ \begin{bmatrix} 0 & 6 \\ & 0 \end{bmatrix} \end{array} \begin{array}{l} (O_2, O_4) \\ (O_1, O_3, O_5) \end{array}$$

- And finally, the algorithm stops when we have all observations in one group.

THE DENDROGRAM



EXAMPLE: PALMER PENGUINS



CONSIDERATIONS

- Single linkage tends to favour “elongated” clusters, while complete linkage favours “round” clusters.
- The algorithms are sensitive to outliers.
- Greedy algorithms: if an observation is misallocated at the beginning of the procedure there is no way to correct it.
- Computationally complex.

PARTITIONAL CLUSTERING

- An alternative to hierarchical clustering is the class of partitional methods.
- Given a number of groups (decided *a priori*), these methods look for a **partition** of the observations into distinct non-overlapping groups.
- We will focus on *k-means* clustering, but other methods exist, such as the more robust Partition Around Medoids (PAM).
- The network methods that we will discuss next can also be considered partitional methods.

K-MEANS

Given a set of *n* cells and *k* groups, *k-means* looks for a partition of the data with an iterative algorithm.

Starting from an initial set of *k* centroids, at each iteration:

- We assign each cell to the nearest centroid
- We re-compute a new set of centroids

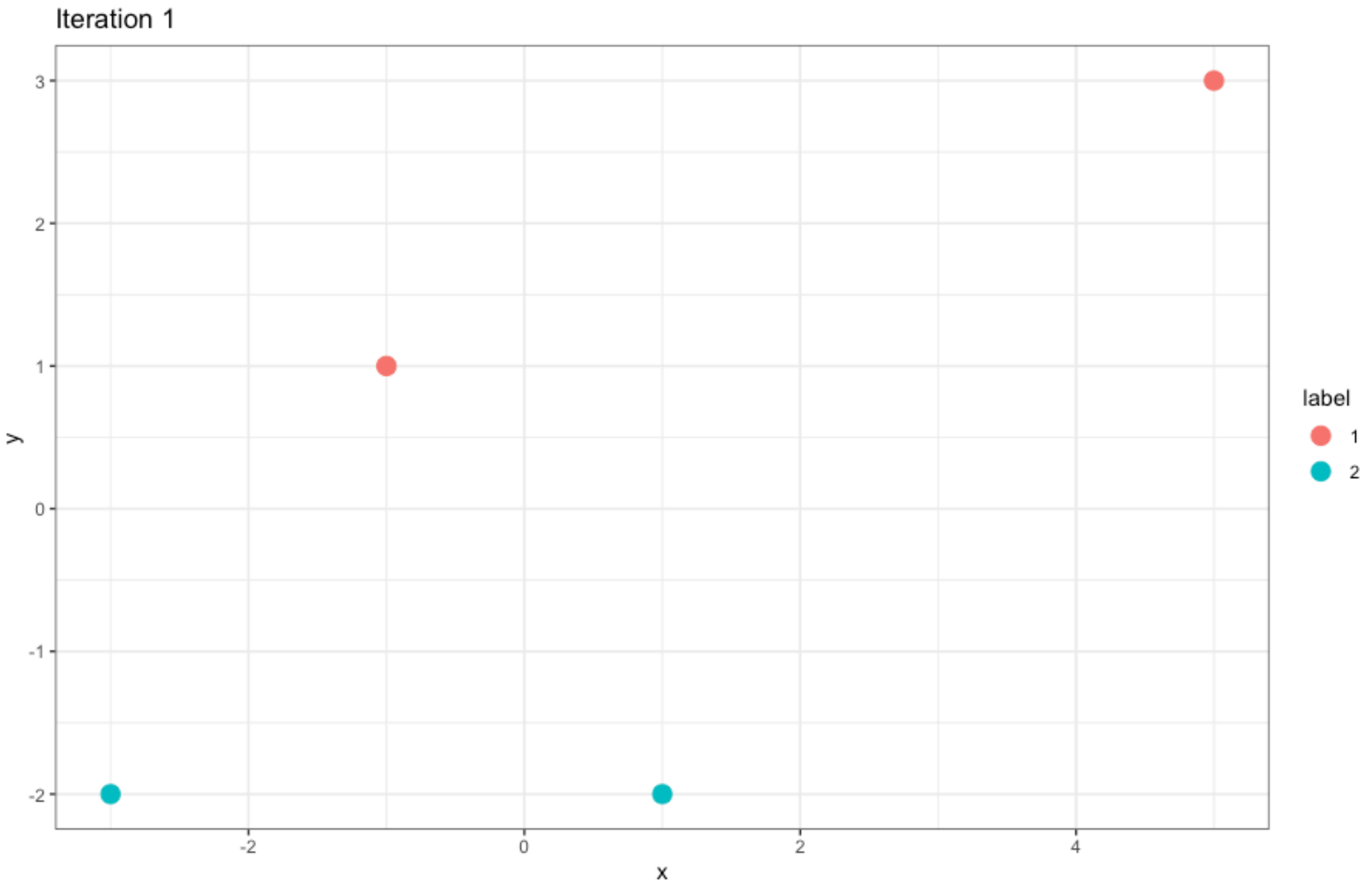
K-MEANS: TOY EXAMPLE

- Assume we have two variables and four observations:

	x_1	x_2
A	5	3
B	-1	1
C	1	-2
D	-3	-2

- At the beginning of *k-means* we have to choose the number of groups and an initial partition, say (AB) vs (CD) ($k = 2$).

K-MEANS: TOY EXAMPLE



K-MEANS: TOY EXAMPLE

	x_1	x_2
A	5	3
B	-1	1
C	1	-2
D	-3	-2

- we compute the centroids of each group, simply as the average of the coordinates.

$$c_{(AB)} = \left(\frac{5+(-1)}{2}, \frac{3+1}{2}\right) = (2, 2)$$

$$c_{(CD)} = \left(\frac{1+(-3)}{2}, \frac{-2+(-2)}{2}\right) = (-1, -2)$$

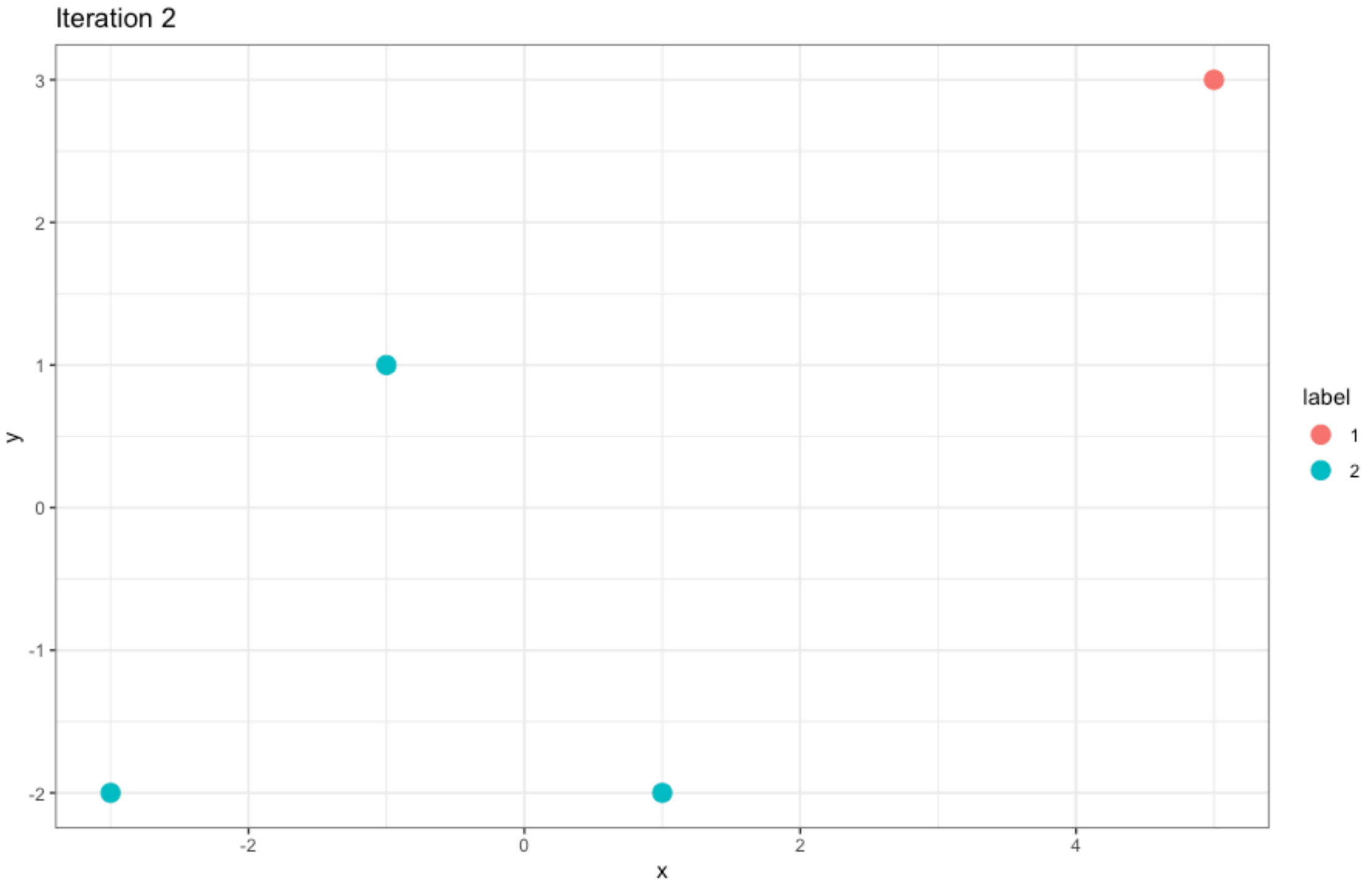
K-MEANS: TOY EXAMPLE

- We now compute the distance between each observation and each centroid.

	$c_{(AB)}$	$c_{(CD)}$
A	$\sqrt{10}$	$\sqrt{61}$
B	$\sqrt{10}$	$\sqrt{9}$
C	$\sqrt{17}$	$\sqrt{4}$
D	$\sqrt{41}$	$\sqrt{4}$

- And assign each observation to the nearest centroid.
- Namely, A to the first centroid and B, C, D to the second.

K-MEANS: TOY EXAMPLE



K-MEANS: TOY EXAMPLE

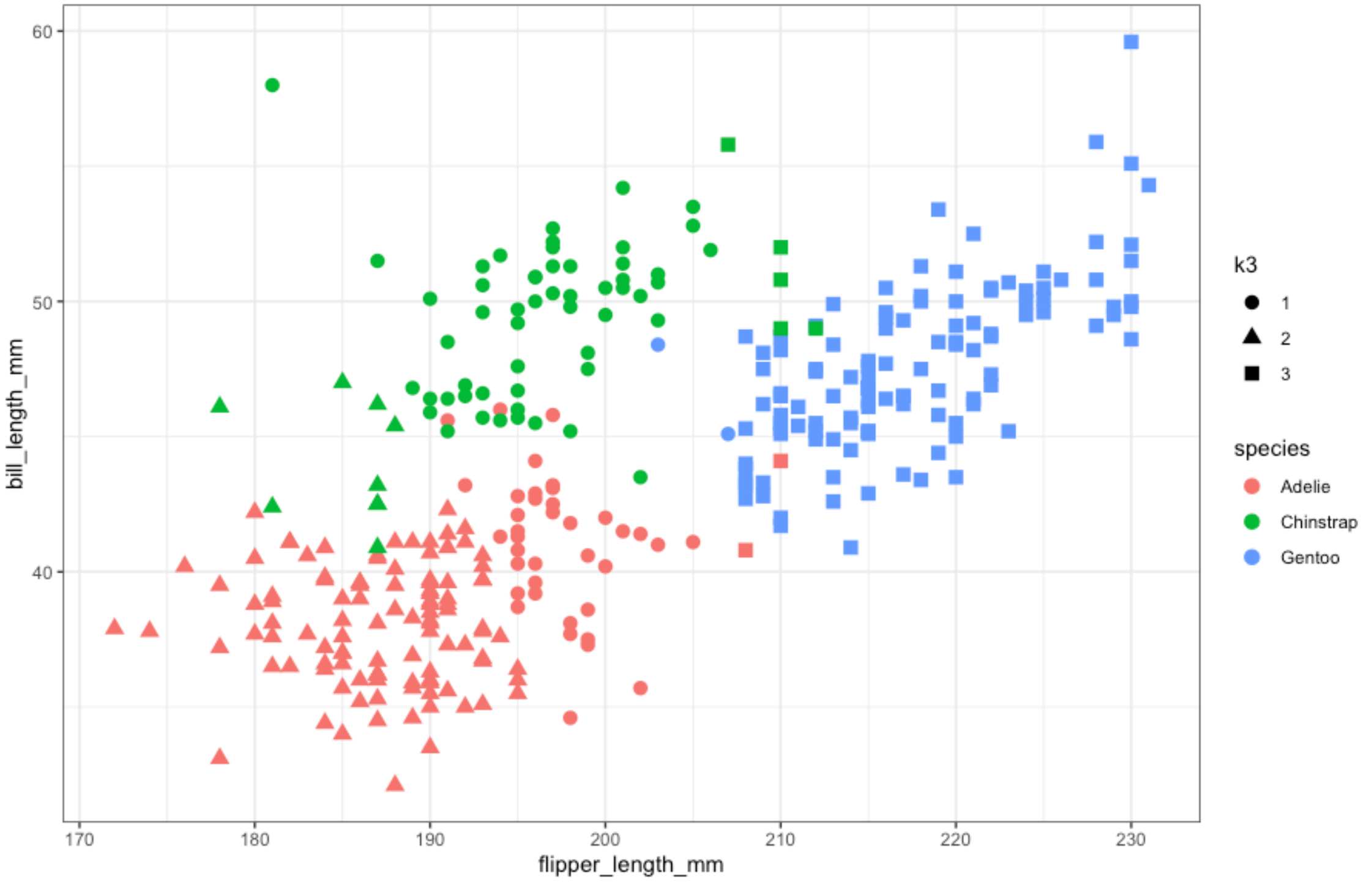
- We re-compute the centroids of each group and the new distances from the centroids.

$$
\begin{aligned}
c_{(A)} &= (5, 3) \\
c_{(BCD)} &= \left(\frac{-1+1+(-3)}{3}, \frac{1-2+(-2)}{3}\right) = (-1, -1)
\end{aligned}
$$

	$c_{(A)}$	$c_{(BCD)}$
A	$\sqrt{0}$	$\sqrt{52}$
B	$\sqrt{40}$	$\sqrt{4}$
C	$\sqrt{41}$	$\sqrt{5}$
D	$\sqrt{89}$	$\sqrt{1}$

- No points change group membership, hence the algorithm stops.

EXAMPLE: PALMER PENGUINS



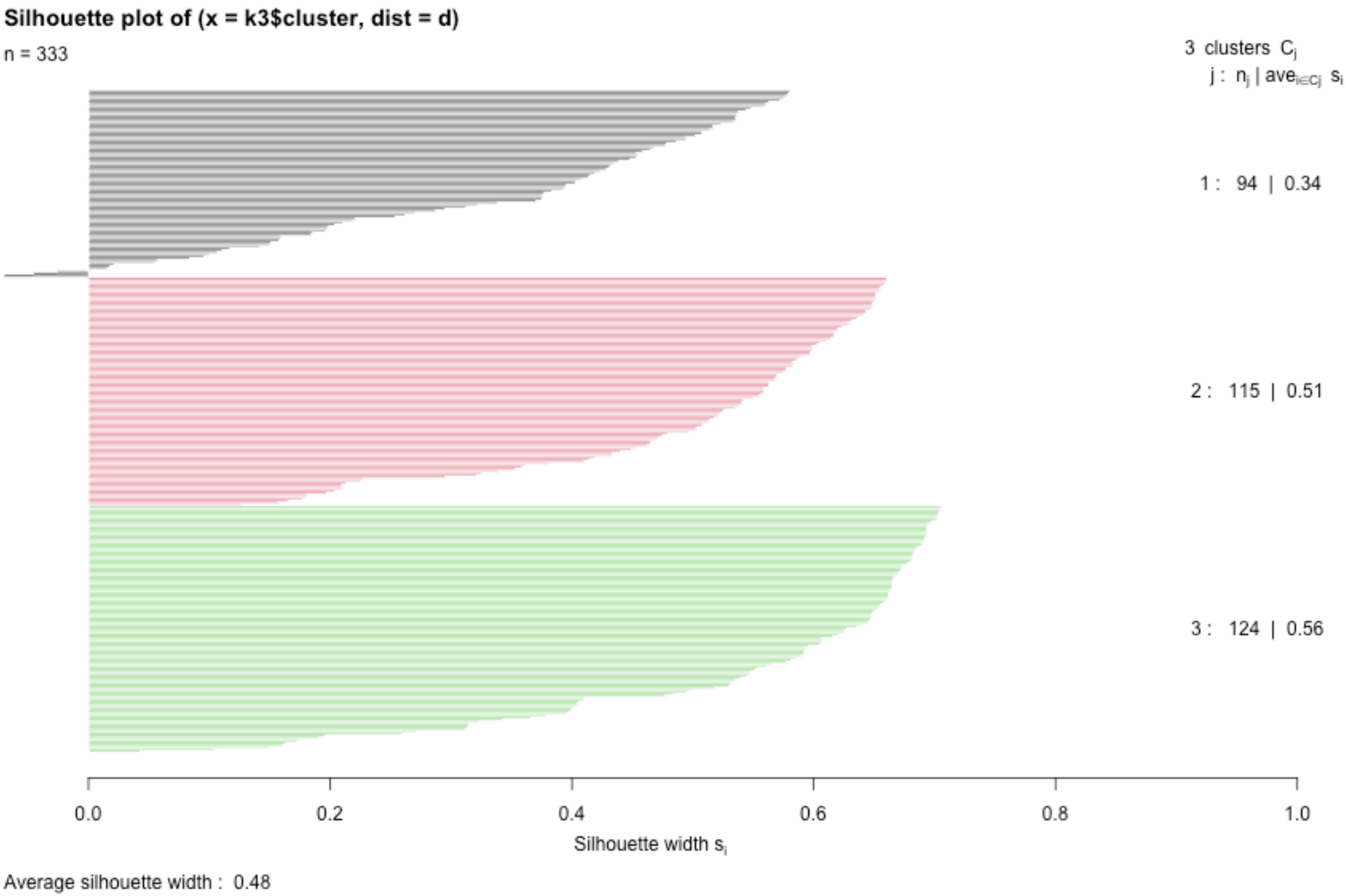
PERFORMANCE EVALUATION

- We ask for groups, the algorithms return groups. How do we know they represent true variability in the data?
- One useful measure is the **silhouette index** defined as

$$sil(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \in [-1,1]$$

- Where $a(i)$ is the average distance between observation i and the other observations in the same cluster;
- $b(i)$ is the minimum distance between observation i and the observations in the other clusters.

EXAMPLE: PALMER PENGUINS



NETWORK-BASED METHODS

First, we construct a **graph** (network), in which nodes are observations and they are connected to the nearest neighbors.

To avoid the curse of dimensionality, the graph is created using a reduced dimensional space (e.g., PCA).

We then use specific algorithms to identify **communities** of cells that are highly interconnected in the graph.

Each community represents a cluster.

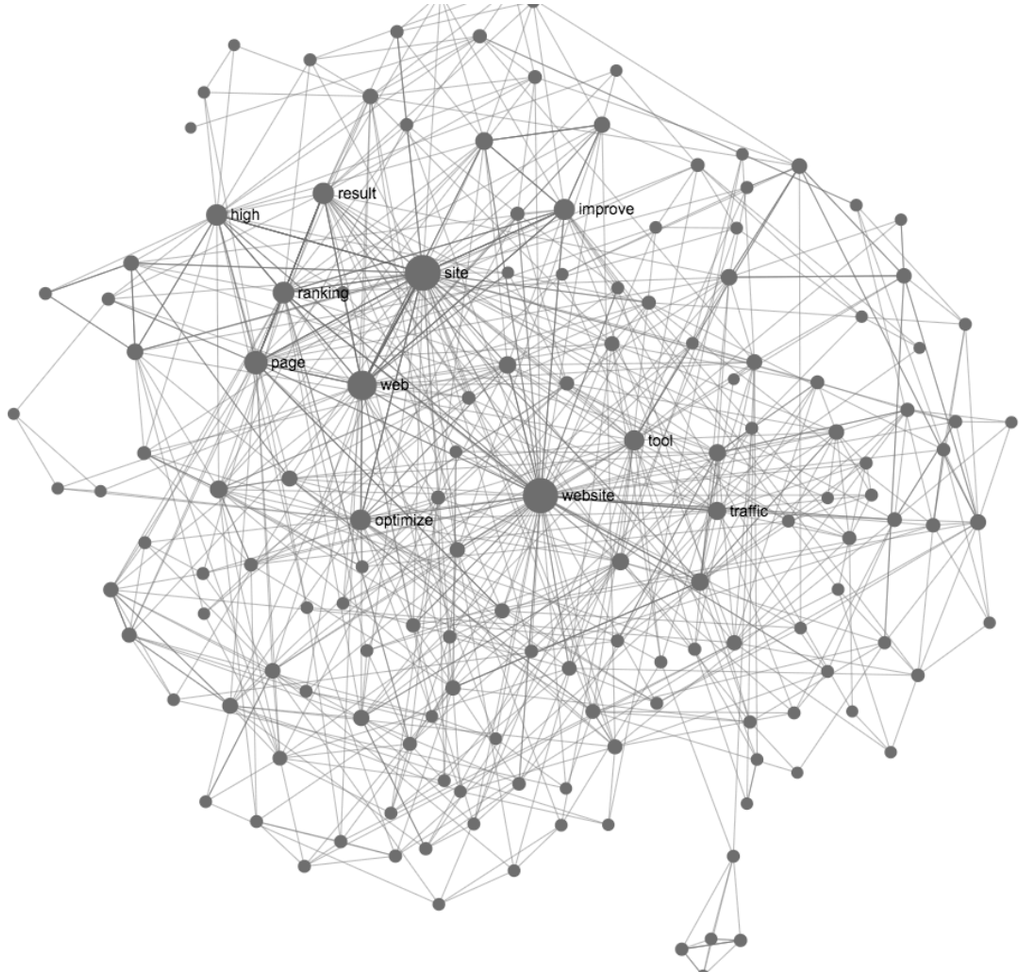
NETWORK-BASED METHODS

The rationale is that there are groups of observations highly connected to each other but with few connections to observations outside the community.

A network has a good **community structure** if there is a high proportion of connections within the communities and a low proportion of connections between the communities.

EXAMPLE: COMMUNITY STRUCTURE

Good community structure



Poor community structure

COMPARISON WITH K-MEANS

The major advantage of this class of methods is its **scalability**.

These methods only require to search for the *k* nearest neighbors and there are highly efficient algorithms for this task.

The main drawback is that, once the network is built, there is no other information about the samples that is used to perform the clustering.

IMPLEMENTATION

As in any algorithm, we need to set some parameters:

- How many neighbors (*k*) to consider when creating the graph.
- How to weight the edges of the graph.
- How to define the communities, starting from the graph.

GRAPH CONSTRUCTION

The graph construction is carried out with the following steps.

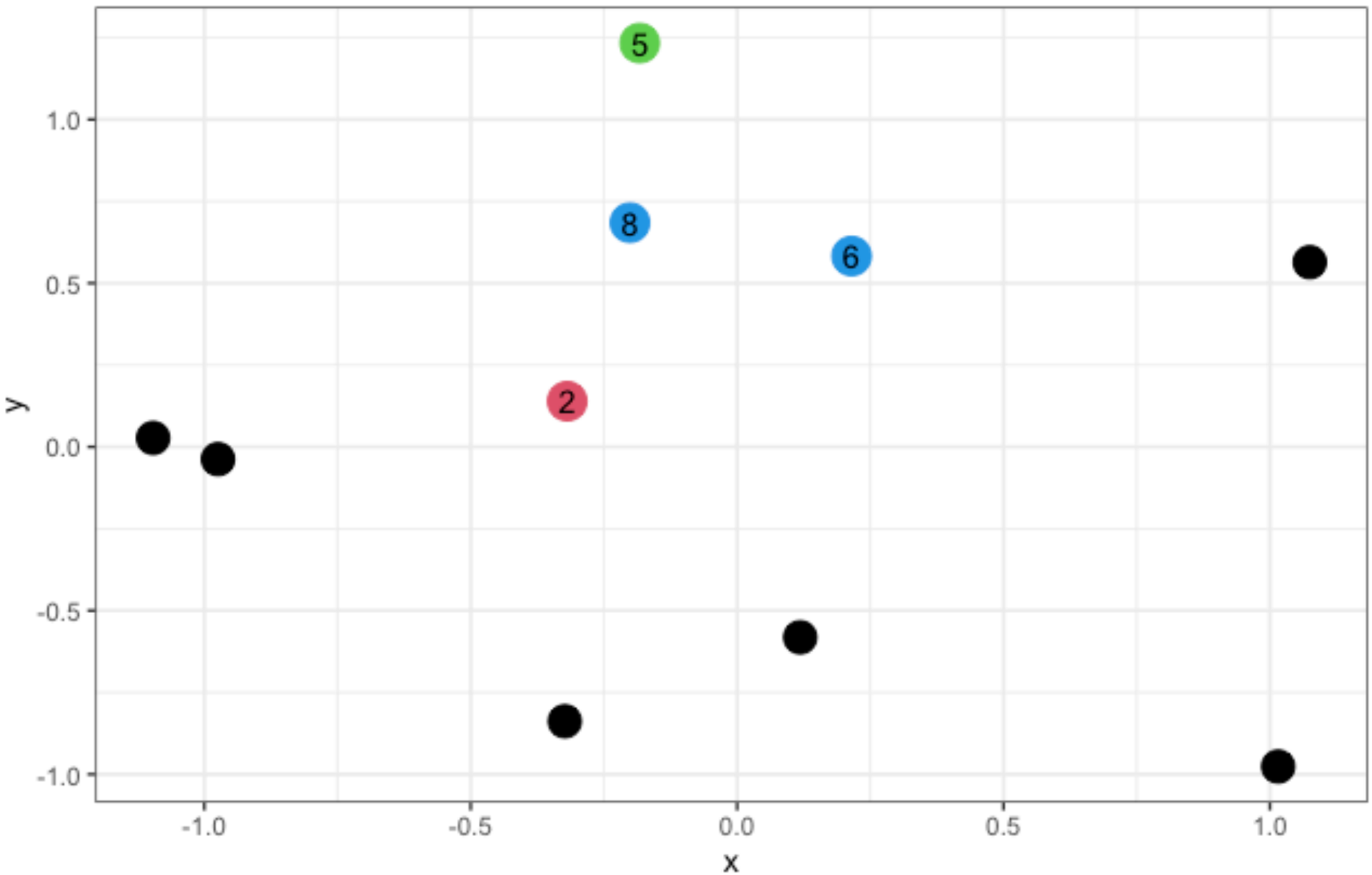
- For each observation, we identify the *k* nearest neighbors, based on the Euclidean distance in an appropriate space.
- An edge is added for each pair of observations that have at least one of their *k* neighbors in common, weighted by how similar they are.

Similarity is given by $k - r/2$, where r is the smallest sum of the ranks of the two nearest nodes.

For instance, if observation i is the nearest to both j and l, then the weight between j *and* l is $k - 1$.

GRAPH CONSTRUCTION: TOY EXAMPLE

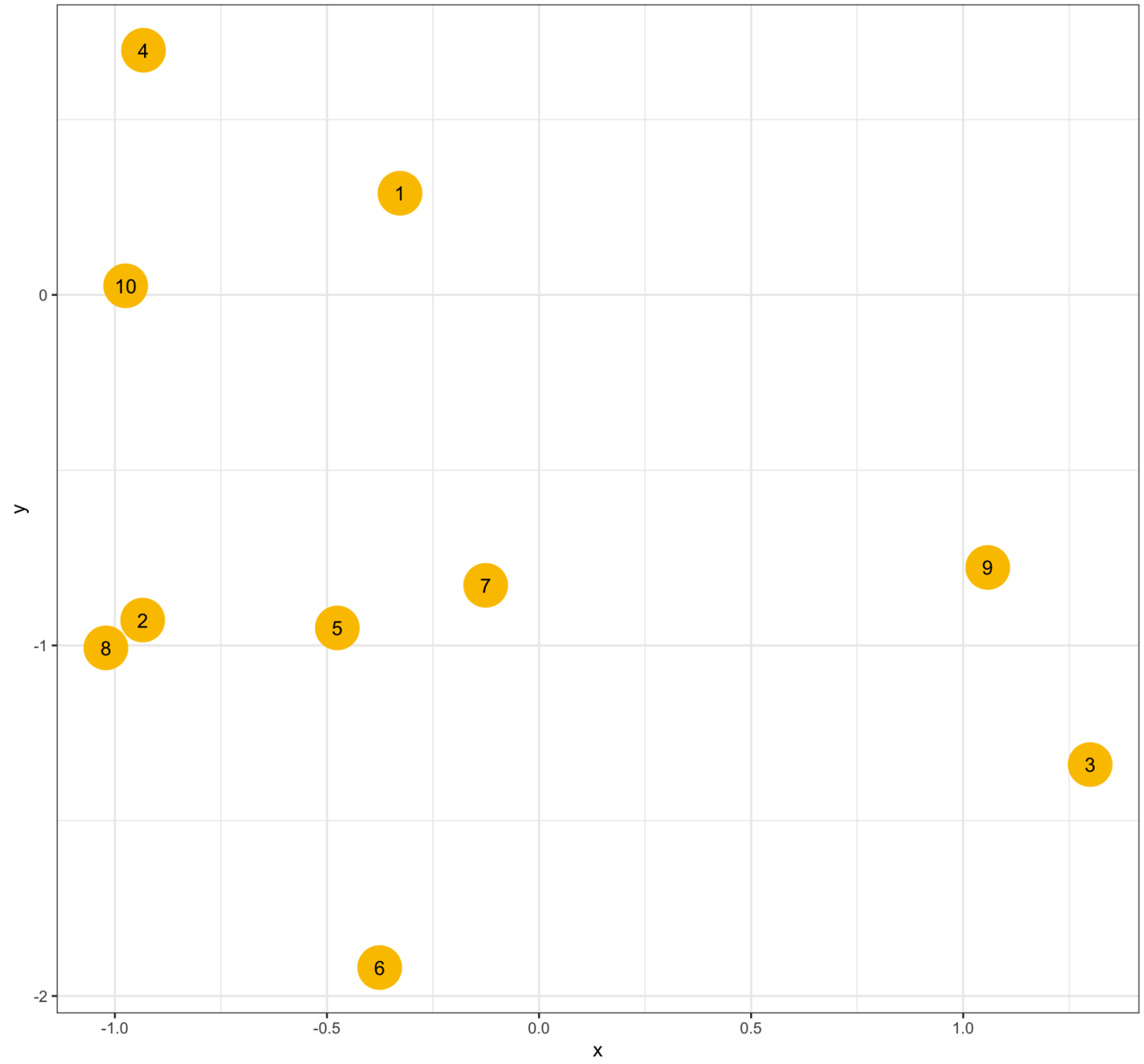
We want to see whether x_2 and x_5 should be connected in the graph.

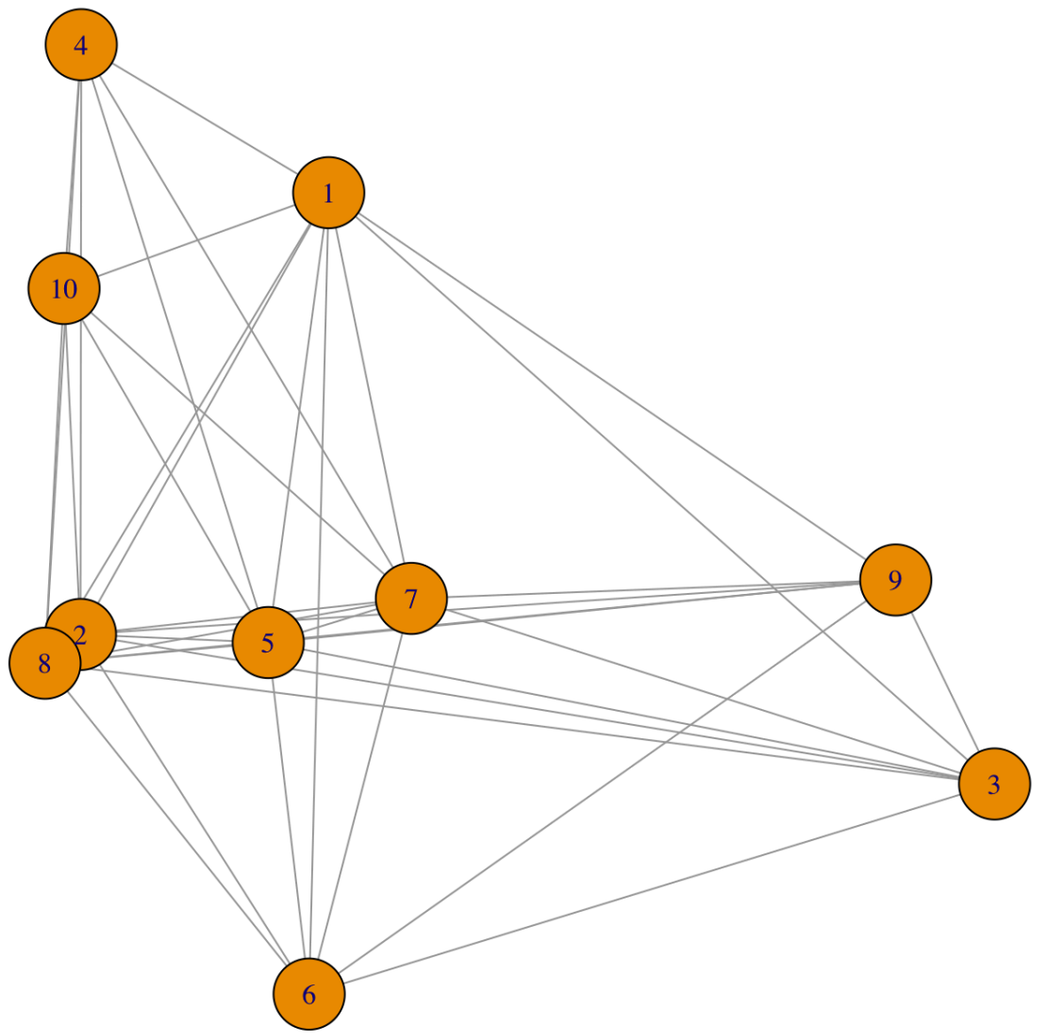


GRAPH CONSTRUCTION: TOY EXAMPLE

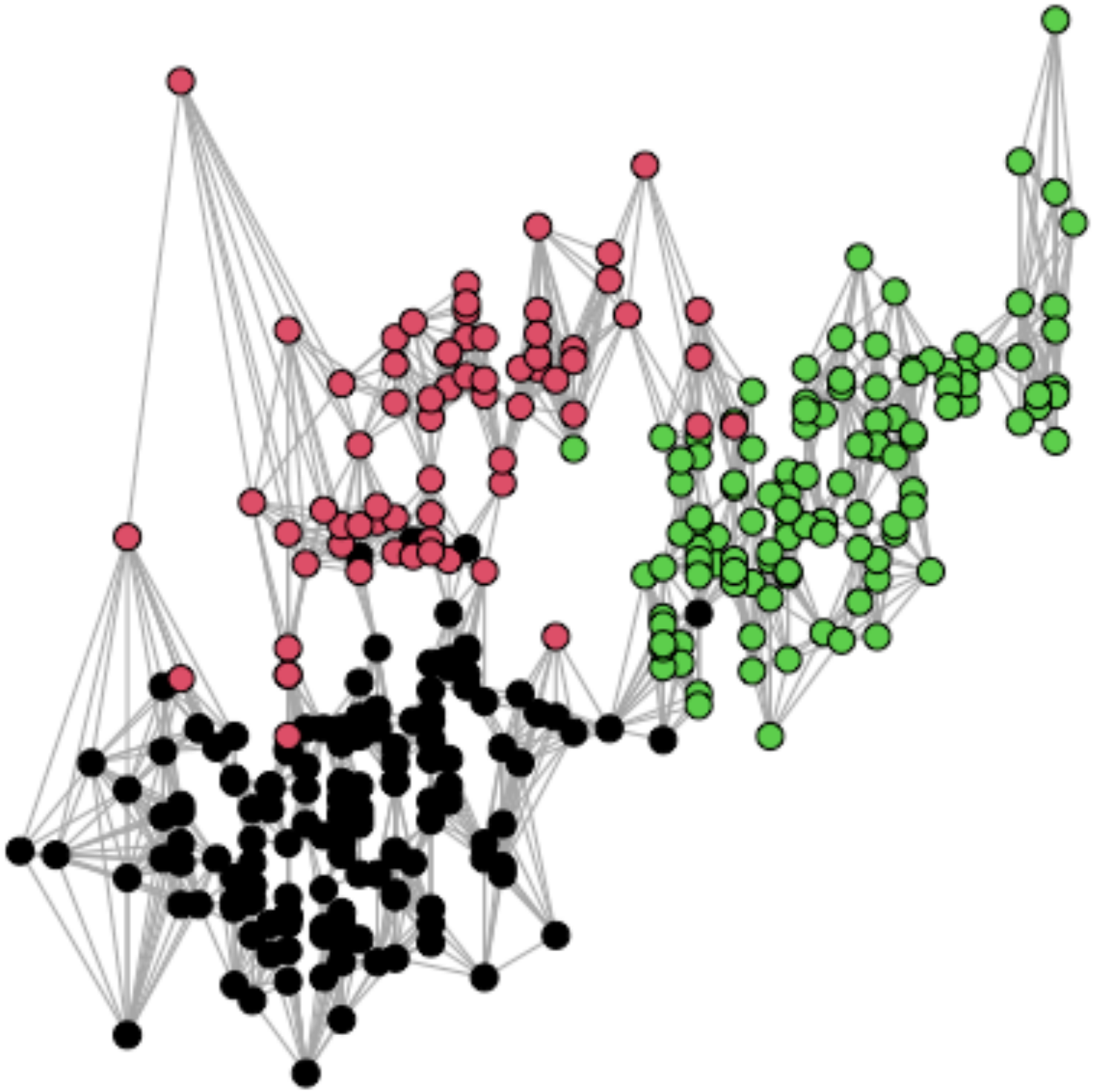
- Imagine that we choose $k = 3$
- x_6 and x_8 are among the 3 nearest neighbours of both x_2 and x_5.
- In particular, x_8 is the nearest neighbour of both (rank 1).
- Hence, the similarity between x_2 and x_5 is $k - r/2 = 3 - (1 + 1)/2 = 2.$

GRAPH CONSTRUCTION: TOY EXAMPLE





GRAPH CONSTRUCTION: PENGUINS

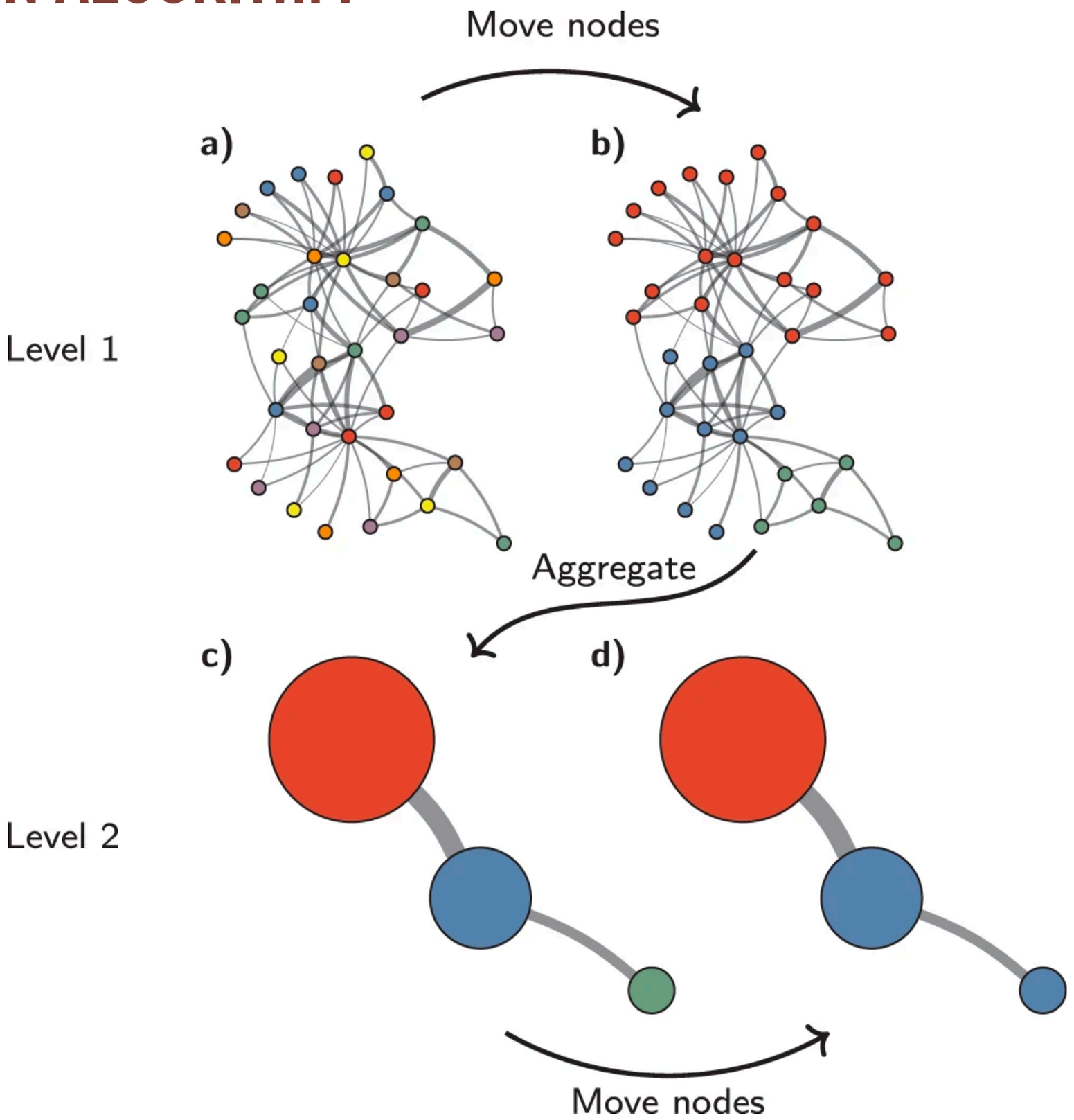


THE LOUVAIN ALGORITHM

The goal is to maximise the **modularity** of the network, with two steps that are repeated until the modularity cannot grow anymore:

1. Nodes are moved from one community to another.
2. Nodes are aggregated into a community and a new network is created using communities as nodes.

THE LOUVAIN ALGORITHM



https://www.nature.com/articles/s41598-019-41695-z

MODULARITY

Modularity is a measure of how much a network is separated into communities.

It is computed as the difference between the observed number of edges and the expected number in each community:

$$Q = \frac{1}{2m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(c_i, c_j),$$

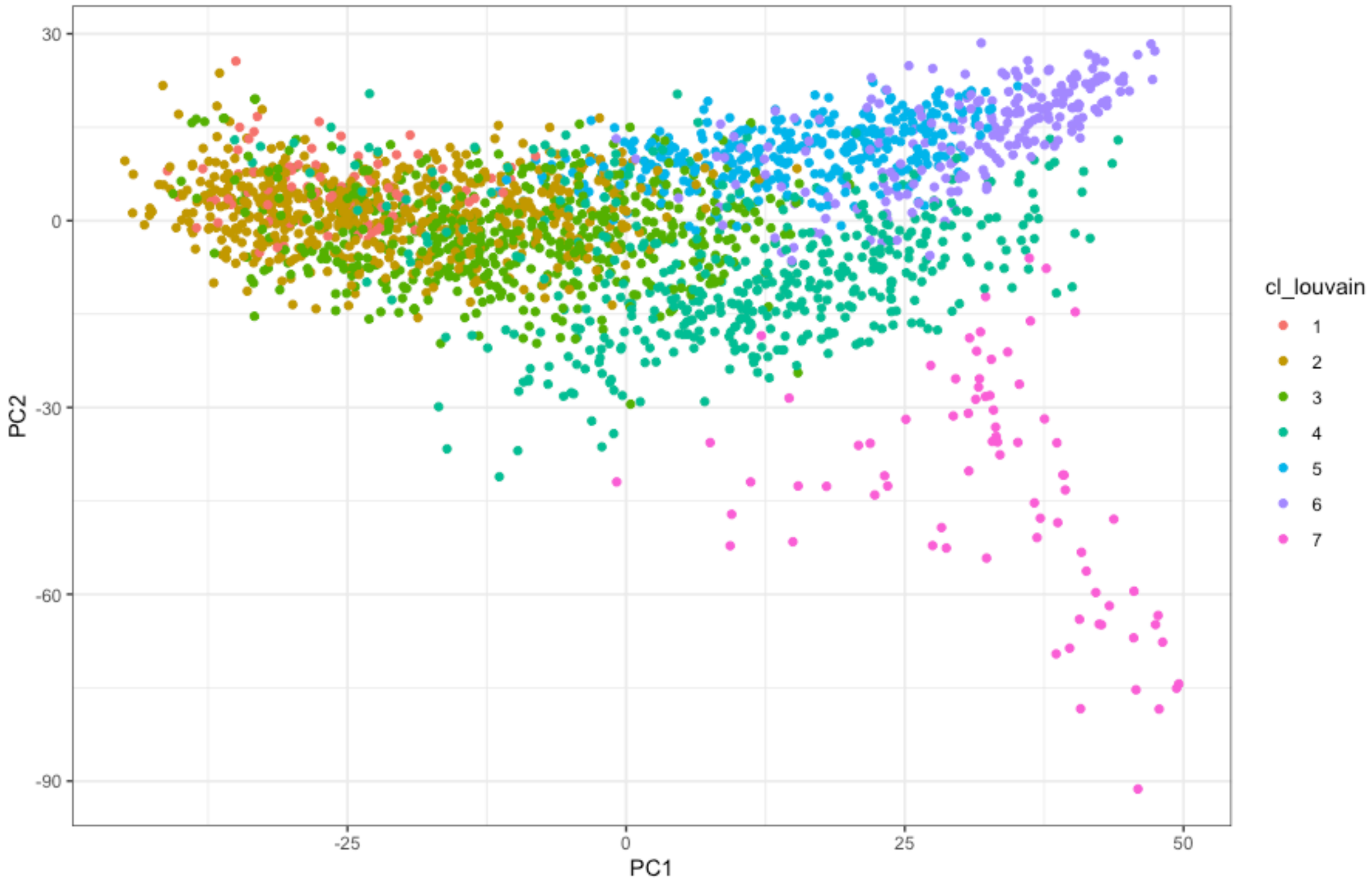
High values of modularity indicate a high community structure.

THE WALKTRAP ALGORITHM

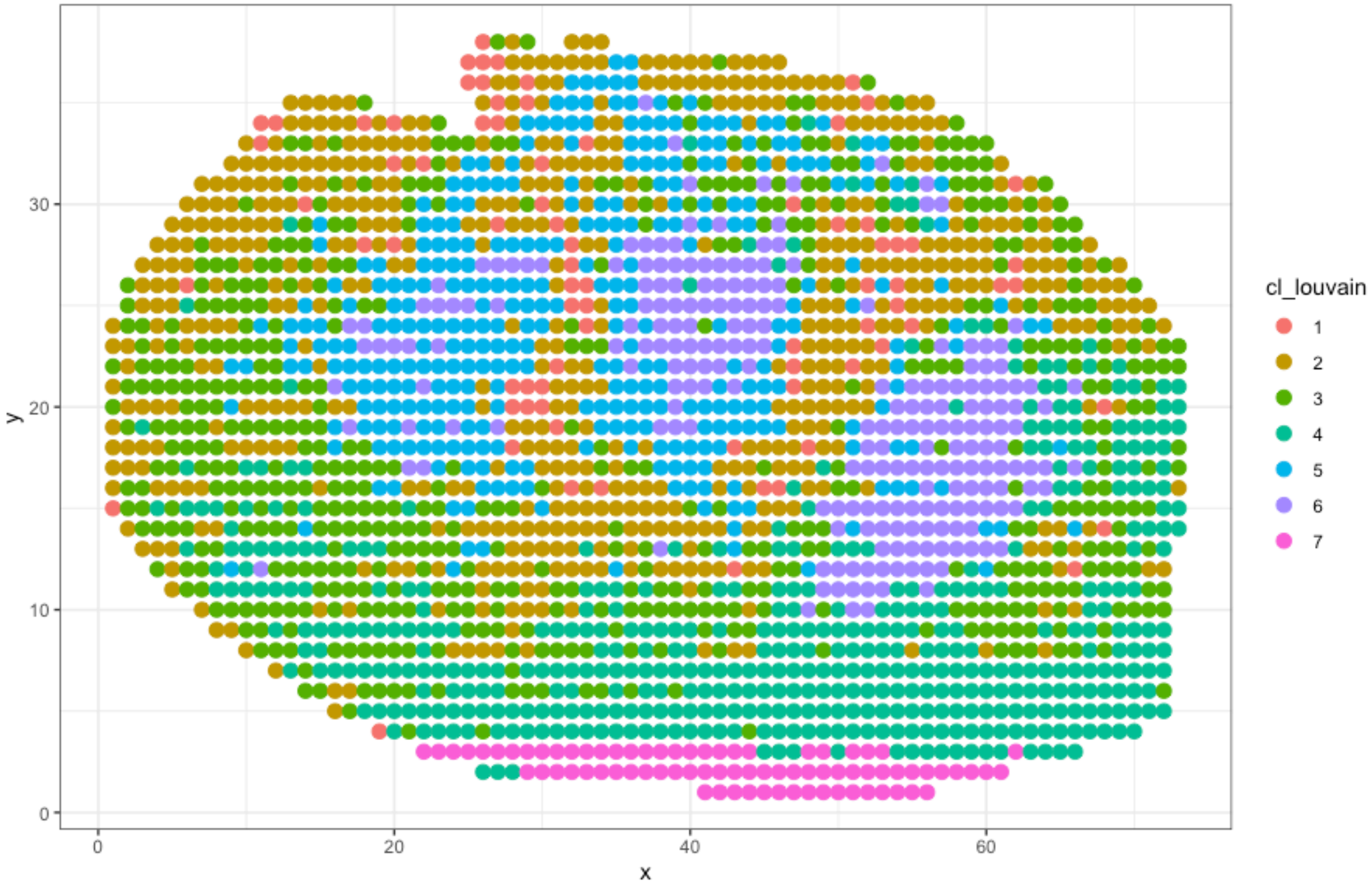
Another method is to use a **random walk** on the graph: the rationale is that "walking" randomly along the graph, we will get "trapped" into a highly connected community.

Intuitively, the probability to go from one node to another in the network is higher if the two nodes belong to the same community then if they belong to two different communities.

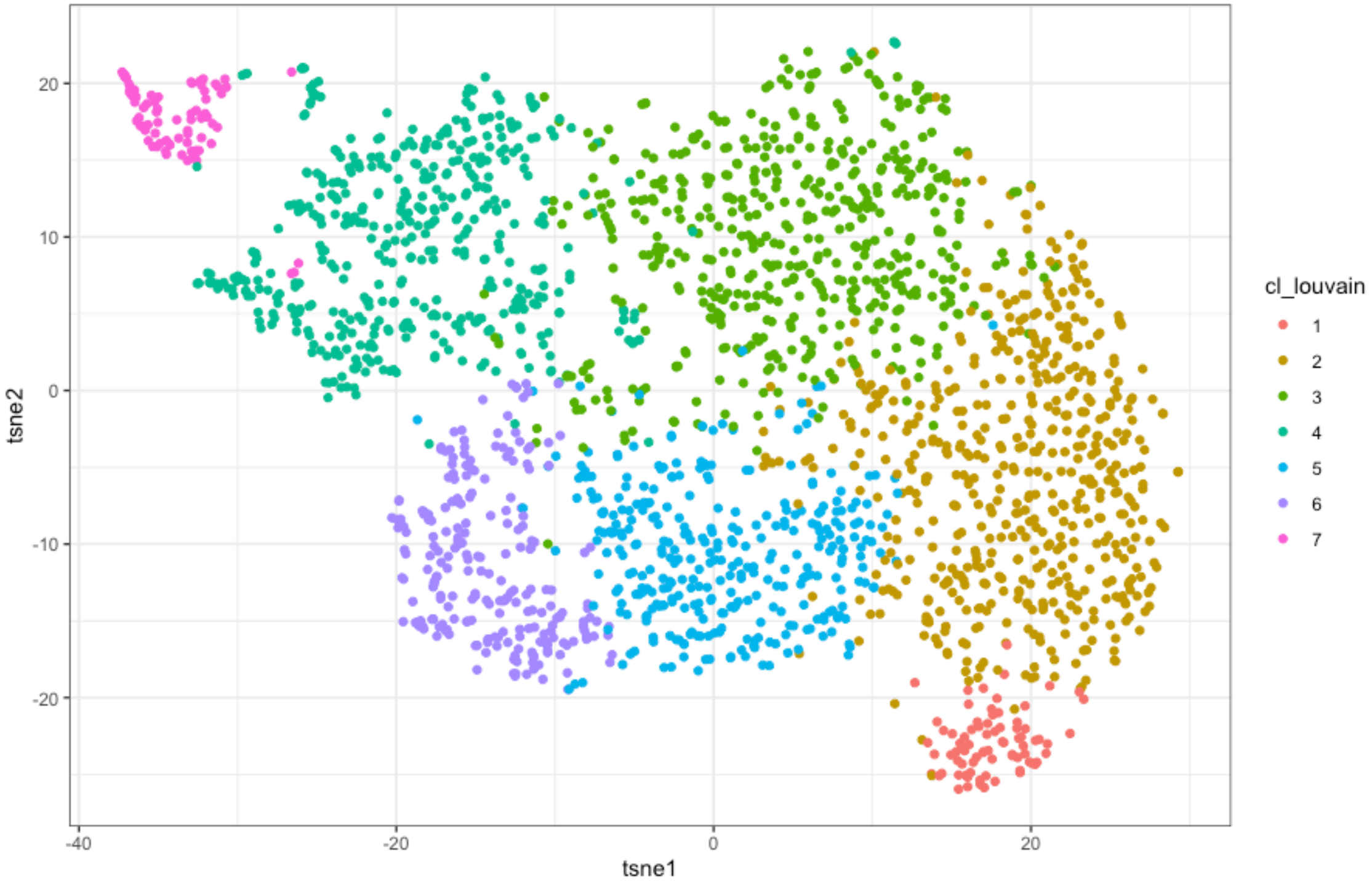
NYAKAS (2013) DATA



NYAKAS (2013) DATA



NYAKAS (2013) DATA



WHAT TO DO WITH COVARIATES?

- Again, think about the biological question that we are trying to answer and the design of our experiment.
- If our additional variables represent a signal that we want to include in the low-dimensional representation, we can include them in the clustering process e.g. using the K prototype algorithm (***clustMixType*** package).
- If our additional variables represent nuisance factors that we want to account for in the representation, we can add them as covariates in the dimensionality reduction step and cluster the samples in that low-dimensional space.

SUPERVISED ANALYSES

SUPERVISED ANALYSIS

- When we talk about *supervised analysis*, we typically refer to two distinct statistical problems.

1. We might want to identify the differentially expressed proteins between two or more groups (say, healthy vs. diseased or treated vs. controls)

2. We might want to leverage the protein expression profiles of each patient to predict their phenotype/ outcome (e.g., subtype of disease, prognosis, …).

SUPERVISED ANALYSIS

- In the first case, we can translate the question into a statistical hypothesis testing.
- In the second case, we may want to use a model to predict a *response* variable given the set of *predictors* (protein levels).
- It turns out that both approaches can be stated in terms of a regression model.
- We will start from the simple case of testing the difference between two groups and build up to more complex, penalized regression approaches.

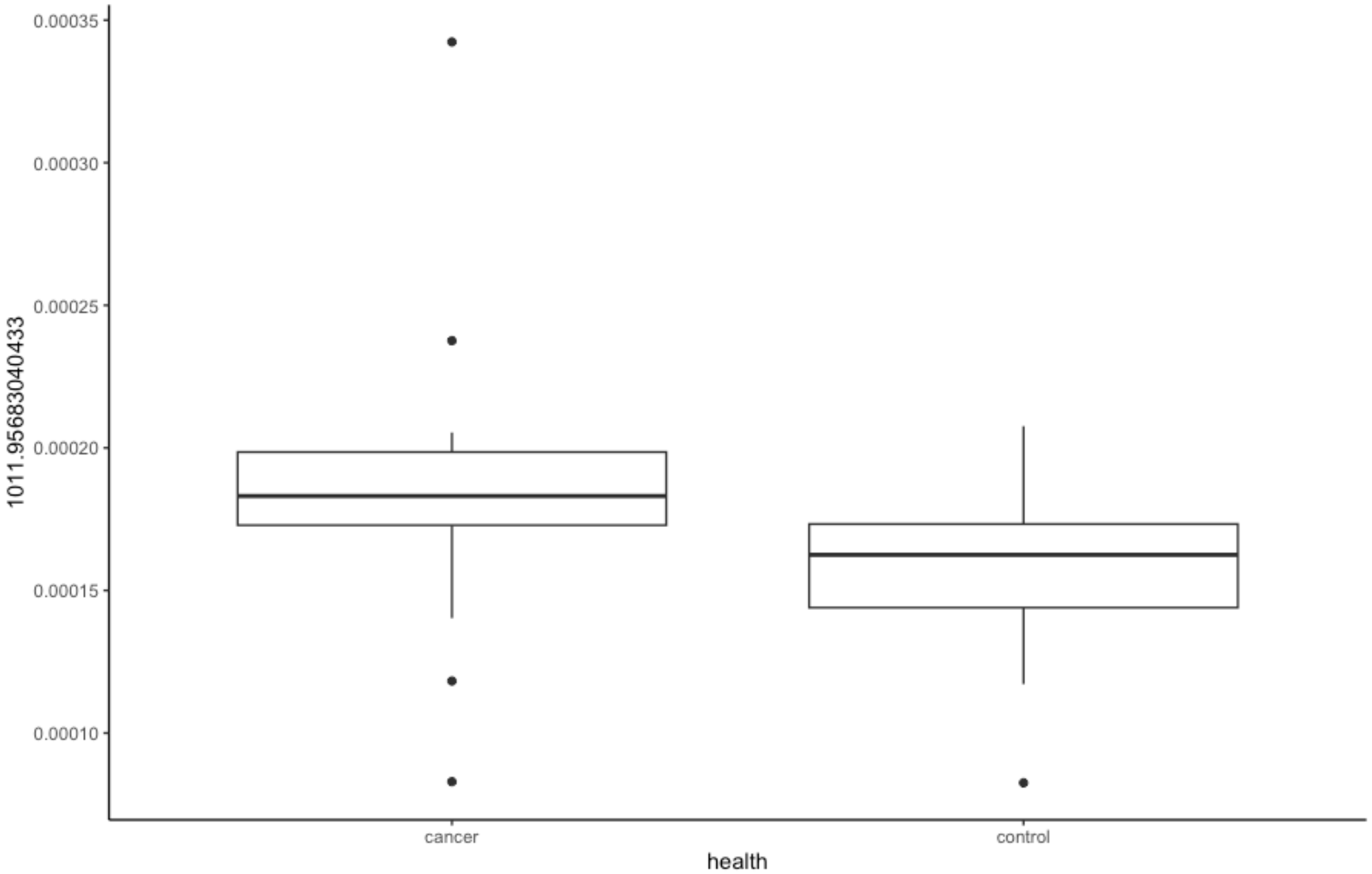
THIS IS JUST A TEASER…

- We would need a full semester course to properly teach these topics.
- Here, we will just give you some pointers and intuitions (+ R examples).
- I encourage you to read more about this on your own (see last slide).

COMPARING TWO GROUPS

- Imagine that we have only one protein.
- Imagine that we want to compare the level of that protein between two groups.
- Let's consider the first protein of the Fiedler et al. (2009) data.
- We want to compare its expression between cancer and control groups.

COMPARING TWO GROUPS



HYPOTHESIS TESTING

- Is the *average expression* the same or different between the two groups?
- In statistical terms, this is called a hypothesis testing problem.
- We want to contrast two hypotheses, known as the *null hypothesis* (H_0) and the *alternative hypothesis* (H_1), respectively.
- *H_0: the mean of the protein expression is equal between the two groups.*
- *H_1: the mean of the protein expression is different between the two groups.*

ARE THEY DIFFERENT?

- Mean cancer: 1.85
  Mean control: 1.58

- Is this difference *significant*?

- In other words, is this difference a random feature of this particular dataset or is it a general characteristic of the *population*?

- If there was no real difference and we repeated the experiment a large number of times, say 10,000 times, would we expect to observe this difference *simply by chance*?

LET'S "SIMULATE" CHANCE BY SHUFFLING LABELS

	y	x
1	1.66	control
2	1.86	control
3	1.87	control
4	1.69	control
5	3.42	cancer
6	1.74	cancer
7	2.05	cancer
8	1.93	cancer

D = 0.515

	y	x
1	3.42	control
2	1.69	control
3	1.74	control
4	1.66	control
5	1.86	cancer
6	1.87	cancer
7	2.05	cancer
8	1.93	cancer

D = -0.2

	y	x
1	1.87	control
2	1.93	control
3	1.66	control
4	1.69	control
5	1.74	cancer
6	3.42	cancer
7	2.05	cancer
8	1.86	cancer

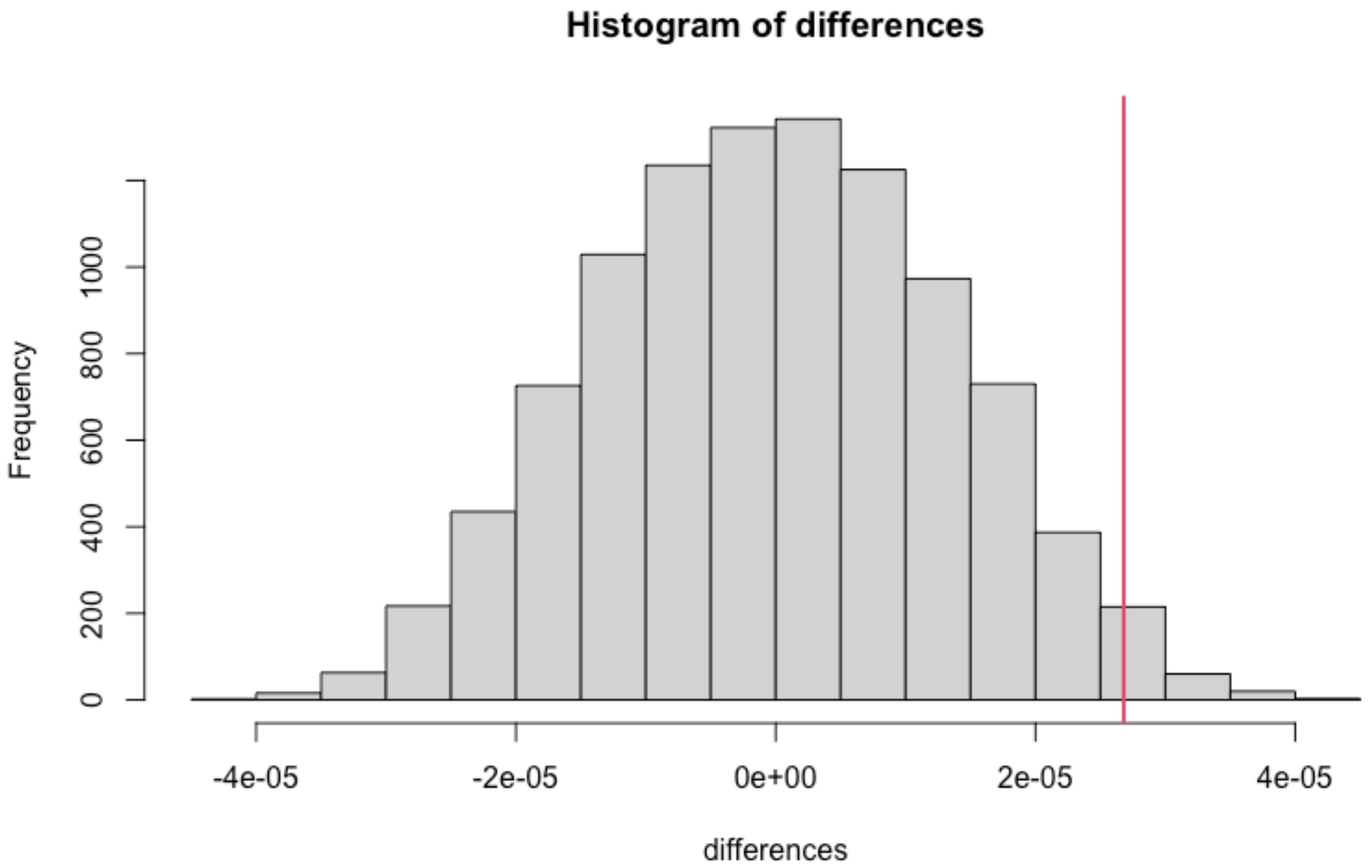
D = 0.48

	y	x
1	1.93	control
2	1.86	control
3	2.05	control
4	1.66	control
5	1.74	cancer
6	1.87	cancer
7	1.69	cancer
8	3.42	cancer

D = 0.305

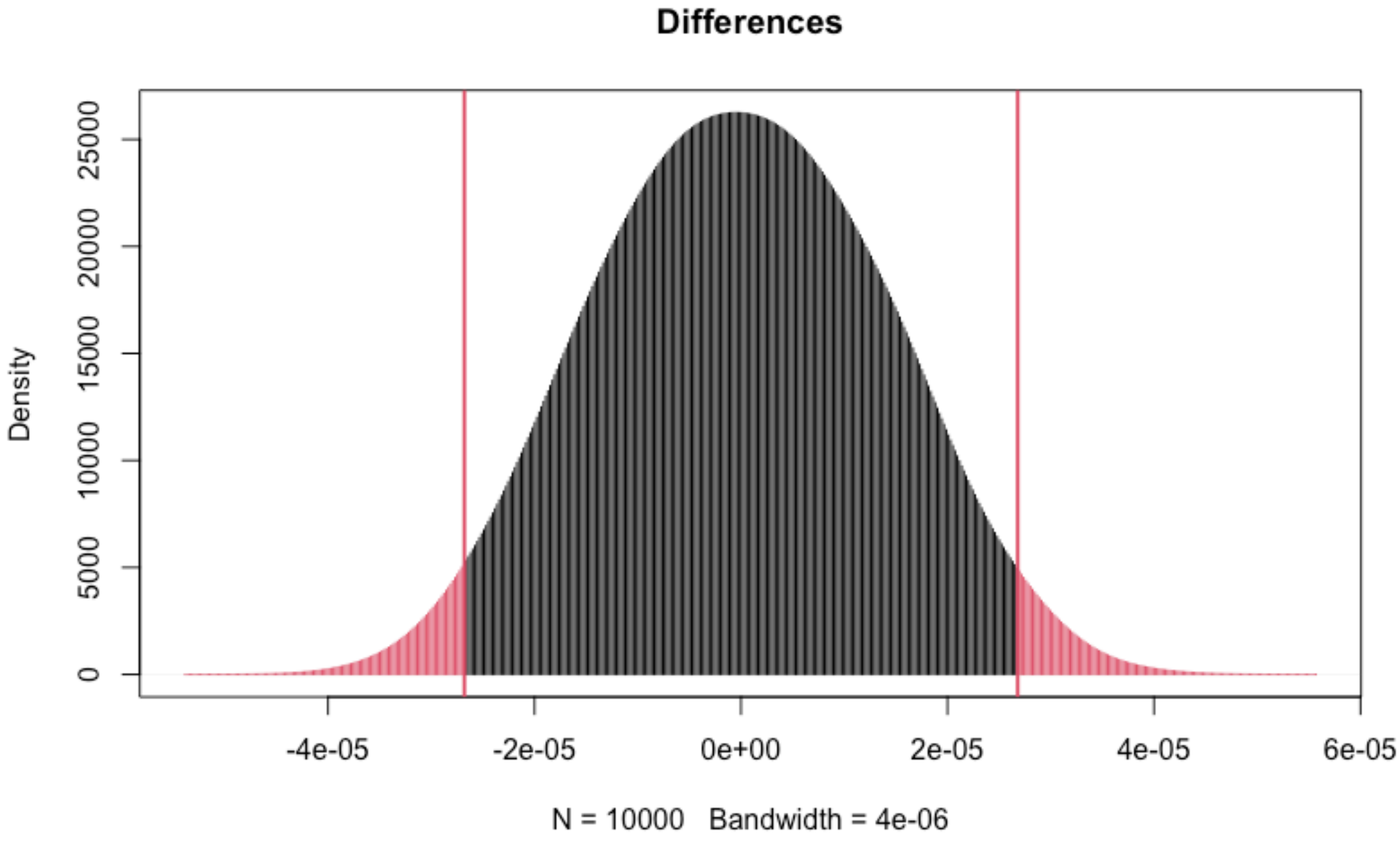
...

HOW MANY TIMES DO WE OBSERVE A MORE EXTREME RESULT?



- Observed difference vs distribution of differences by chance.

P-VALUE



- Probability of obtaining a result as far or farther from H_0 than the observed outcome.

THE T-TEST

- It turns out that if we adjust the difference of the mean by an appropriate quantity (and with some assumptions) we know the distribution expected under H_0.
- This is the widely used *t-test*.
- We reject the null hypothesis if the p-value is smaller than a certain threshold, typically 0.05.
- This means that the probability of observing such a difference (or bigger) by chance is only 1 in 20 (5%).

EXAMPLE

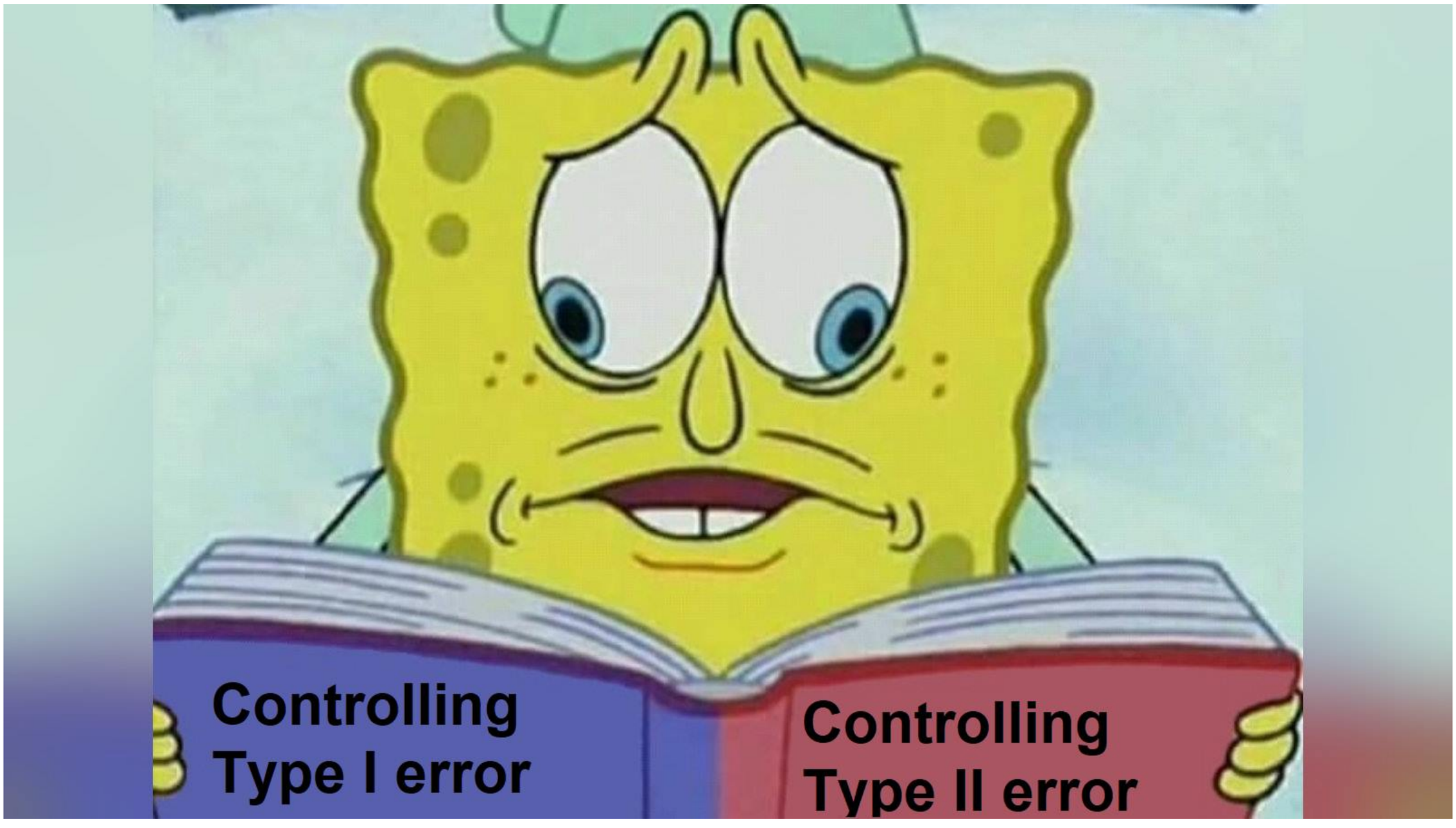
```
        Two Sample t-test

data:  y by x
t = 2.0592, df = 38, p-value = 0.04638
alternative hypothesis: true difference in means between group cancer and group control i
s not equal to 0
95 percent confidence interval:
 4.524305e-07 5.314430e-05
sample estimates:
 mean in group cancer mean in group control
           0.0001851283           0.0001583299
```

- The difference between the two groups is significant.
- (Although only just).

MULTIPLE TESTING

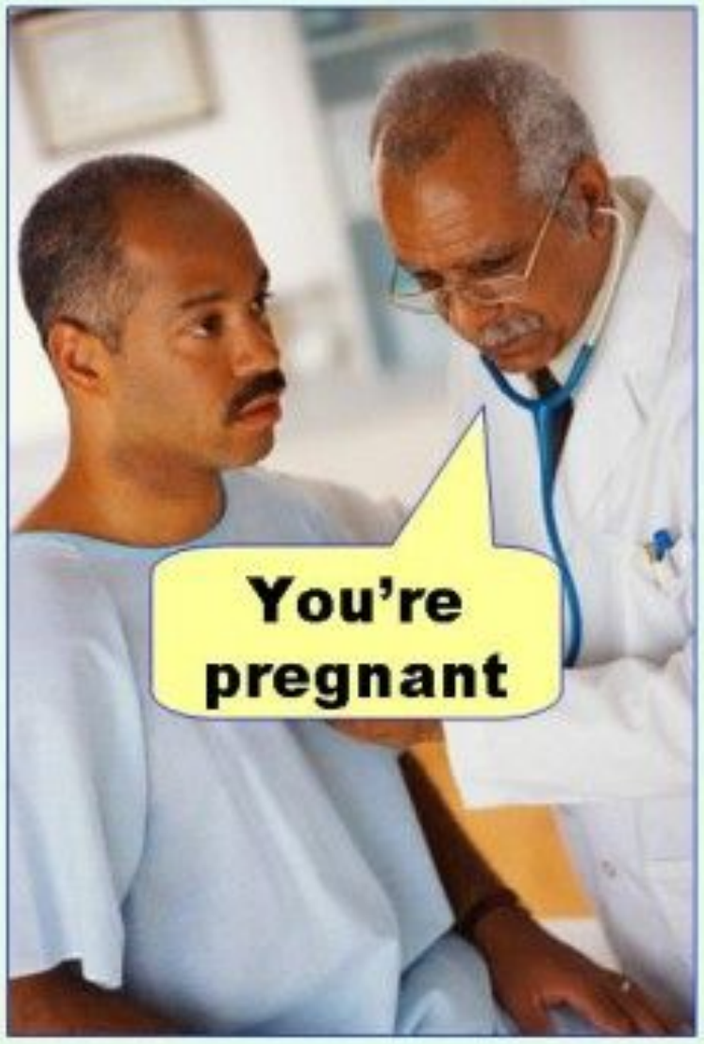
- If we test a hypothesis at a 5% level, we have a 5% probability of rejecting the null hypothesis when it is true.
- This is called the *type I error rate*.
- Now suppose that, instead of only one test, we perform many tests.
- The probability to incorrectly rejecting a null hypothesis increases quite quickly.

INTERLUDE: TYPE I AND TYPE II ERRORS

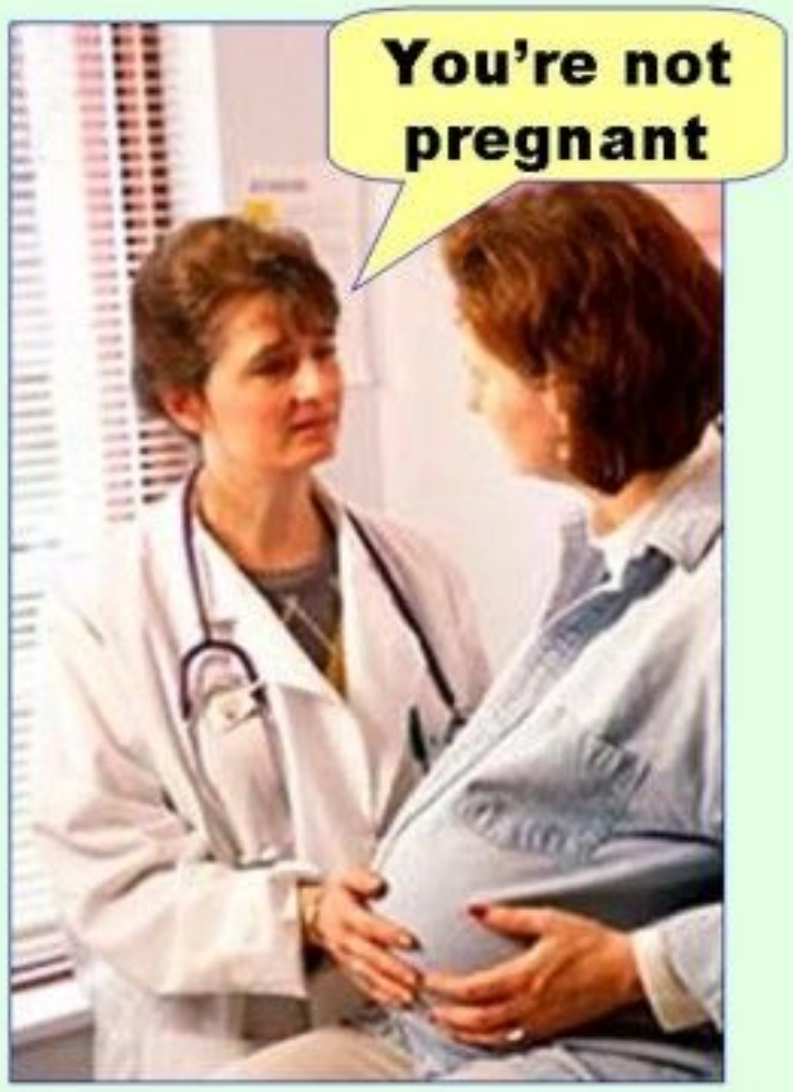


INTERLUDE: TYPE I AND TYPE II ERRORS

Type I error
(false positive)



Type II error
(false negative)



BACK TO MULTIPLE TESTING

- The probability p of obtaining at least one false positive is:
  - 1 protein –> $p = 5\,\%$
  - 10 proteins –> $p \approx 40\,\%$
  - 20 proteins –> $p \approx 64\,\%$
  - 100 proteins –> $p \approx 99\,\%$
- In the kidney dataset we have more than 5000 proteins to test!
- This means that we expect to have about 250 false positive results when we test at a 5% level.

BONFERRONI CORRECTION

- We define *family-wise error rate* (FWER) the probability of observing *at least one false positive* result among all the tested hypotheses.
- There is a rather simple correction, the Bonferroni procedure, that we can use to control the FWER.
- The Bonferroni correction is simply obtained by dividing the level of the test (typically 5%) by the number of tests.
- Or alternatively by multiplying the p-value by the number of tests.

FALSE DISCOVERY RATE (FDR)

- While extremely simple, the Bonferroni correction is often too conservative.
- In practice, we use a procedure that control the *false discovery rate* (FDR), defined as the expected proportion of false discoveries.
- The Benjamini-Hochberg procedure is the simplest and most popular procedure to control the FDR.

BEYOND THE T-TEST

- What if we want to compare more than two groups?
- And if we want to evaluate how proteins change across a continuous variable?
- We can generalize the t-test by using a linear regression model.
- At first it might seem strange, but testing the difference of two groups with a t-test and fitting a linear model with one binary covariate are two equivalent procedures.

EQUIVALENCE BETWEEN T-TEST AND REGRESSION

```
        Two Sample t-test

data:  y by x
t = 2.0592, df = 38, p-value = 0.04638
alternative hypothesis: true difference in me
s not equal to 0
95 percent confidence interval:
 4.524305e-07 5.314430e-05
sample estimates:
 mean in group cancer mean in group control
         0.0001851283          0.0001583299
```

```
Call:
lm(formula = y ~ x)

Residuals:
       Min          1Q      Median          3Q         Max
-1.022e-04 -1.305e-05  1.526e-06  1.341e-05  1.572e-04

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.851e-04  9.202e-06  20.117   <2e-16 ***
xcontrol    -2.680e-05  1.301e-05  -2.059   0.0464 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.115e-05 on 38 degrees of freedom
Multiple R-squared:  0.1004,    Adjusted R-squared:  0.07671
F-statistic:  4.24 on 1 and 38 DF,  p-value: 0.04638
```

LINEAR REGRESSION MODEL

- The goal is to explain / predict the variation of the response variable Y, using k predictors $X_1, X_2, \ldots, X_k$:
  $Y = \beta_0 + \beta_1 X_1 + \ldots + \beta_k X_k + \varepsilon$

- $\beta_0, \beta_1, \ldots, \beta_k$ are unknown parameters (the regression coefficients) which tell us how Y varies when the X's change.

- ε is the unobservable error which expresses the portion of Y that is not explained by the X's.

- In the context of differential expression, we use the expression values as response and the group indicator as predictor.

- Note that we fit one model per protein.

LINEAR REGRESSION FOR DIFFERENTIAL EXPRESSION

- We have only one predictor

$$X = \begin{cases} 0 & \text{if control} \\ 1 & \text{if cancer} \end{cases}$$

- The response variable Y is the protein expression.
- The model is: $Y = \beta_0 + \beta_1 X + \varepsilon$
- Omitting ε, we have that:

$$\begin{cases} \text{If } X = 0 \text{ then } Y = \beta_0 \\ \text{If } X = 1 \text{ then } Y = \beta_0 + \beta_1 \end{cases} \implies \beta_1 = Y_1 - Y_0$$

- where Y_1 is the mean expression in the control and Y_0 is the mean expression in the cancer group.
- Testing the null hypothesis $H_0 : \beta_1 = 0$ is equivalent to test that the means of the two groups are the same.

LINEAR REGRESSION FOR PREDICTION

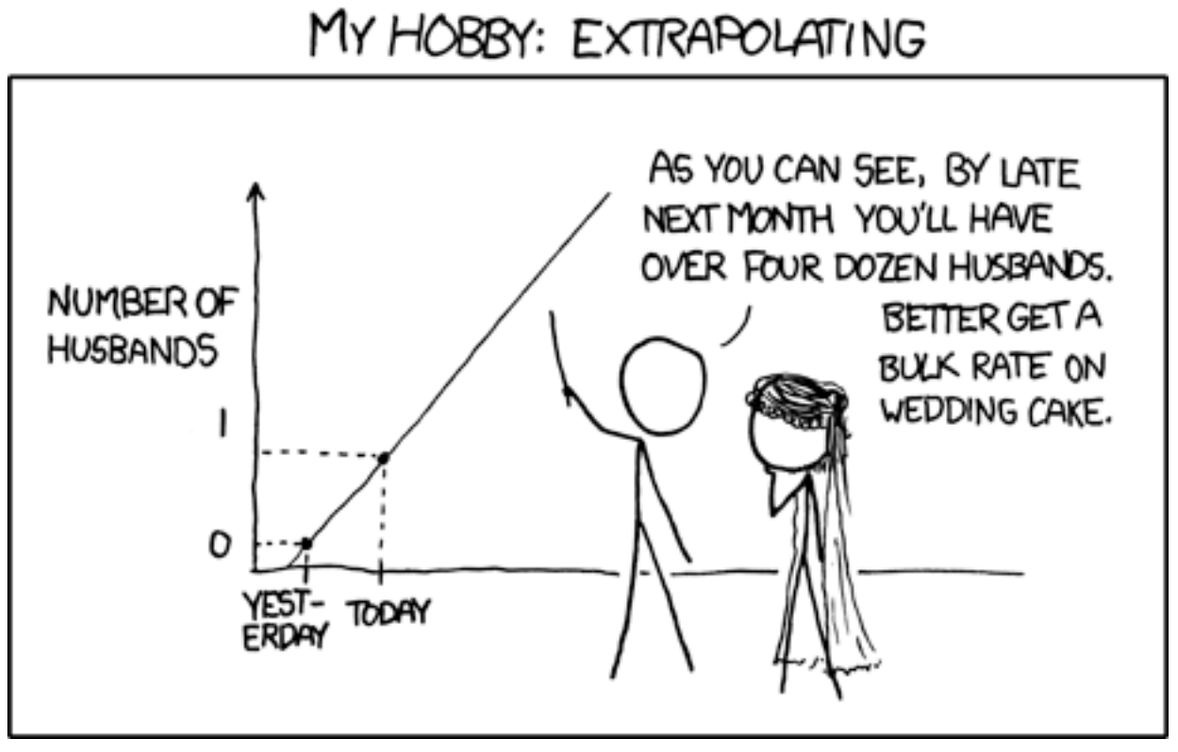
- As said earlier, sometimes the goal is to *predict* the status of a new patient (cancer or control) given their protein expression levels.
- We can use the same model, but switching the roles of response and predictors.
- In this case Y is the binary variable and we include the expression values of all proteins as predictors.
- There are two problems with this:
  - The response Y is binary rather than continuous
  - There are too many predictors (in some cases more than the number of samples!).

LOGISTIC REGRESSION

- In this case, the response variable is binary, e.g.,

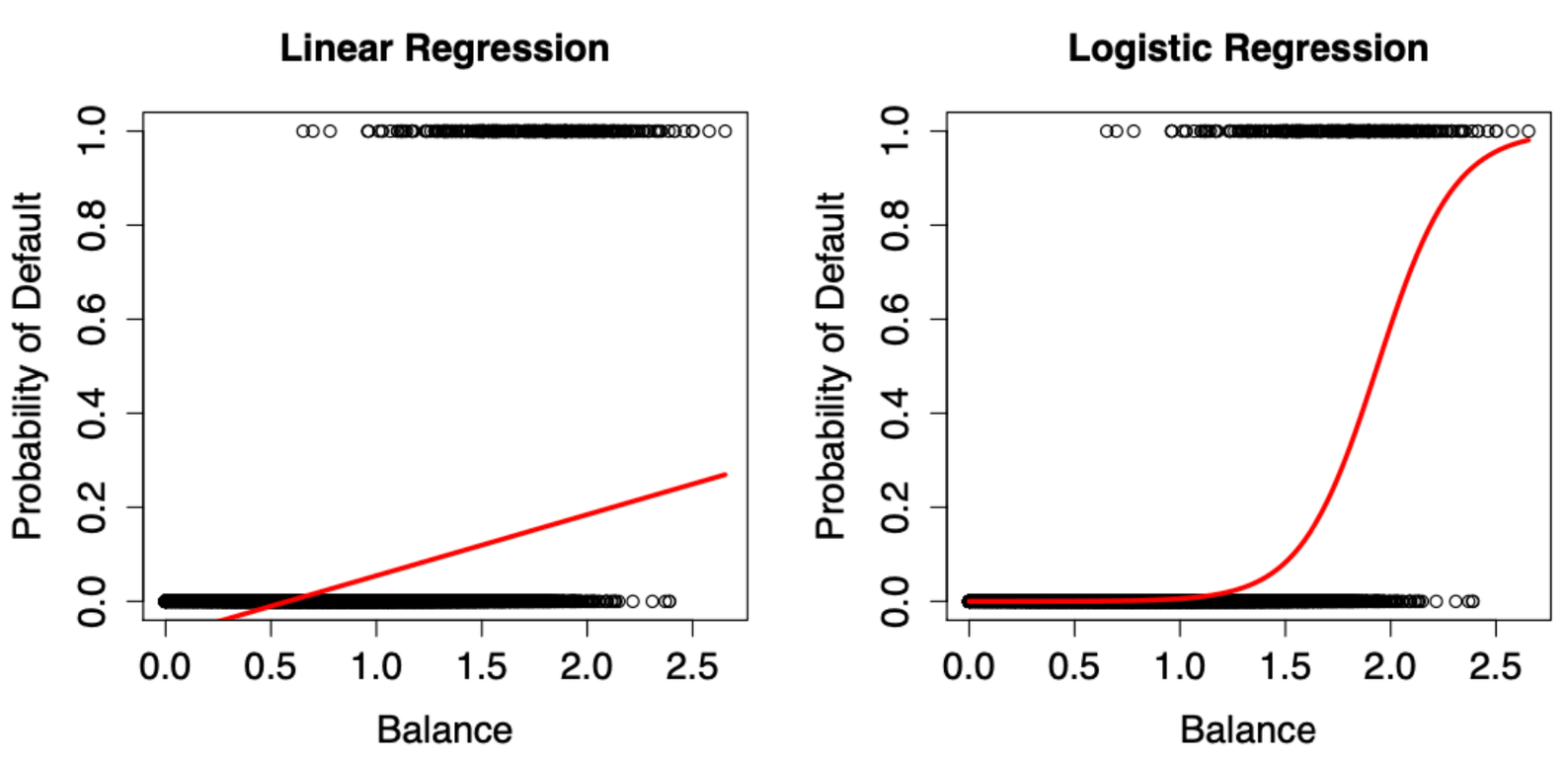
$$Y = \begin{cases} 0 & \text{if control} \\ 1 & \text{if cancer} \end{cases}$$

- We cannot directly model this as a linear combination of the protein levels



LOGISTIC REGRESSION

- logit $P(Y = 1 \mid X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_k X_k + \varepsilon$
- Where $\text{logit}(x) = \log\left(\frac{x}{1-x}\right)$ is the logit function and ensures that we have a probability between 0 and 1.



EXAMPLE: FIEDLER DATA

- Using the first 5 proteins to predict the health status.

```
Call:
glm(formula = y ~ x1 + x2 + x3 + x4 + x5, family = binomial(),
    data = feature_df)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.31298  -0.43258  -0.00247   0.25129   2.94880

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.912e+00  3.299e+00   1.186   0.2358
x1           5.672e+04  5.844e+04   0.971   0.3318
x2           1.466e+04  6.280e+03   2.335   0.0196 *
x3          -1.132e+05  8.931e+04  -1.267   0.2052
x4           6.142e+04  1.106e+05   0.555   0.5789
x5          -9.693e+04  7.836e+04  -1.237   0.2161
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 55.452  on 39  degrees of freedom
Residual deviance: 20.603  on 34  degrees of freedom
AIC: 32.603

Number of Fisher Scoring iterations: 7
```

DID IT WORK?

- The model returns the probability that each observation is “cancer” or “control”.
- We can use a threshold (say 0.5) to classify individuals to the most likely category.

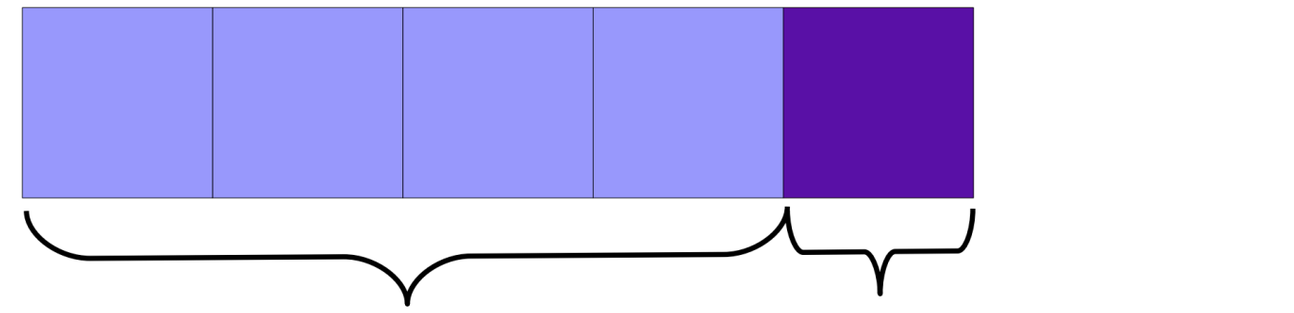
	cancer	control
FALSE	1	18
TRUE	19	2

- The model correctly classifies 37 out of 40 samples.
- However, we are using the same data to train and test the model, risking overfitting.
- A better way is to split the data and use a subset for *training* and another for *testing* the model.

CROSS-VALIDATION

- To avoid overfitting we need to split the data in training and testing.
- A more efficient way is *cross-validation* in which we split the dataset in k groups:
  - Each time we use $k-1$ groups for training and one for testing.
  - We repeat the procedure k times and consider the average performance
- A particular case is the *leave-one-out cross-validation* in which we use $n-1$ observations for training and one for testing.

CROSS-VALIDATION

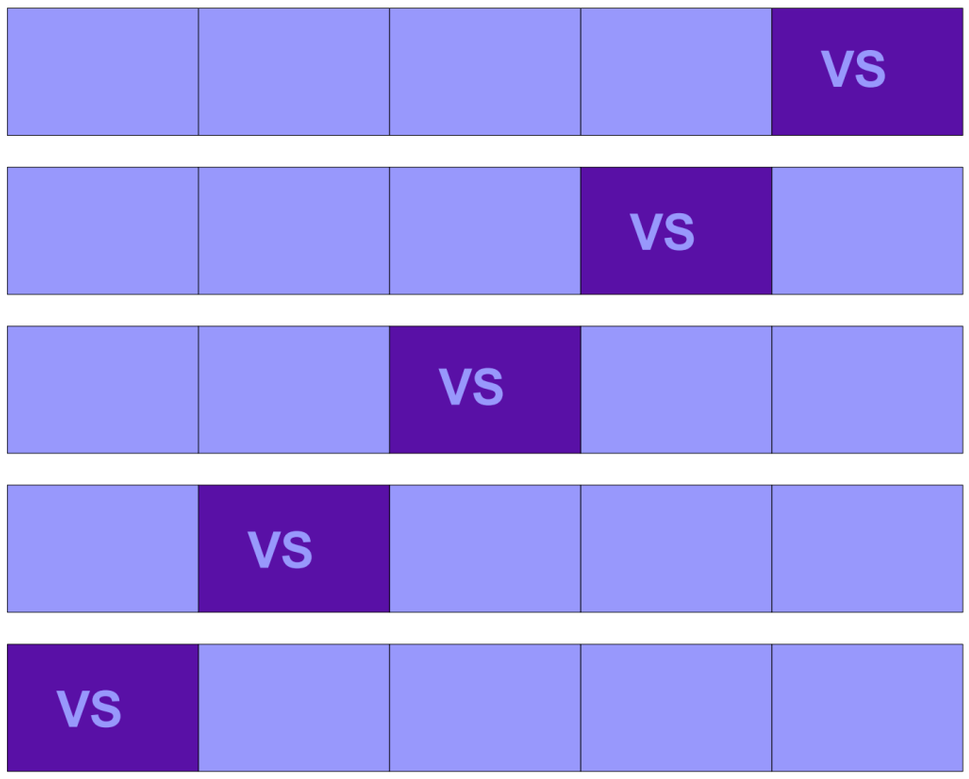


Training set

Train = compute estimator(s)

Validation set

Validate = assess performance of estimator(s)



LINEAR DISCRIMINANT ANALYSIS (LDA)

- In logistic regression we directly model the conditional probability $P(Y|X)$.
- In LDA we model the distribution of X separately in each of the response classes (conditionally on Y) and we use the Bayes' Theorem to obtain $P(Y|X)$.

- If X follows a normal distribution, then results are comparable to those from logistic regression.
- It is possibile to generalise the model to more than 2 classes.
- Many other classification methods exist: QDA, Bayes classifier, support vector machines, random forests, neural networks, ...

HIGH-DIMENSIONAL DATA

- Can we use all the proteins in the model?
- The Fiedler data has 40 observations and 166 proteins.

```
Call:
glm(formula = y ~ ., family = binomial(), data = subset)

Deviance Residuals:
 [1]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
[27]  0  0  0  0  0  0  0  0  0  0  0  0  0  0

Coefficients: (127 not defined because of singularities)
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.218e+02  4.617e+06       0        1
x1          -4.930e+06  1.115e+11       0        1
x2          -1.322e+04  8.414e+09       0        1
x3           7.370e+06  1.061e+11       0        1
x4          -6.352e+05  3.978e+10       0        1
x5           8.521e+06  1.712e+11       0        1
x6          -4.281e+06  8.252e+10       0        1
```

```
x161               NA         NA      NA       NA
x162               NA         NA      NA       NA
x163               NA         NA      NA       NA
x164               NA         NA      NA       NA
x165               NA         NA      NA       NA
x166               NA         NA      NA       NA
```

HIGH-DIMENSIONAL DATA

- Remember that in our case, the number of proteins p is large compared to (often even larger than) the number of observations n.

- In such cases, standard models will not work as we do not have enough observations to estimate all the parameters.

- Possible solutions:
  - Variable selection
  - Regularization/penalization (e.g., Lasso and Ridge).

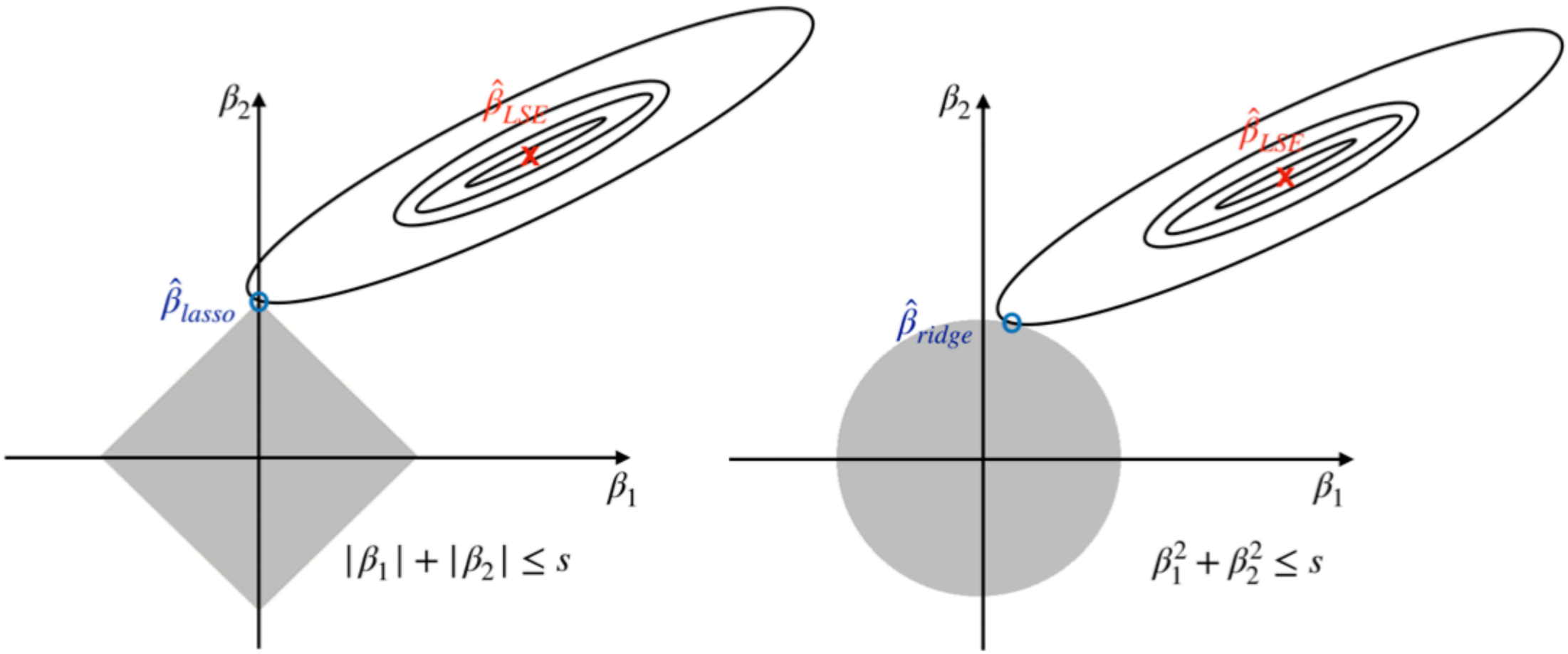
A NAIVE SOLUTION: SELECT THE MOST INFORMATIVE PROTEINS

- A naive (but often effective) solution is to select only the $k(< n)$ most informative proteins to include in the classifier.
- We can achieve this by, e.g., performing a *t*-test for each protein and only keep those significantly differentially expressed.
- It is important that variable selection is included in the cross-validation procedure to avoid re-using the data twice!

AN AUTOMATED PROCEDURE: REGULARIZED REGRESSION

- Penalized (or regularized) regression models allow to include $p > n$ predictors in the regression without need to pre-select them.
- They work by including a penalty term in the computation of the estimates that *shrinks* the values towards 0.
- We have three possibilities depending on which penalty we include in the model.
  - Ridge regression: we penalize by the sum of the squares of the coefficients (L2 norm)
  - Lasso regression: we penalize by the sum of the absolute values of the coefficients (L1 norm)
  - Elastic net: we combine the two penalties together.

RIDGE OR LASSO?



- Lasso shrinks some coefficients exactly to zero, while that's not the case for ridge.

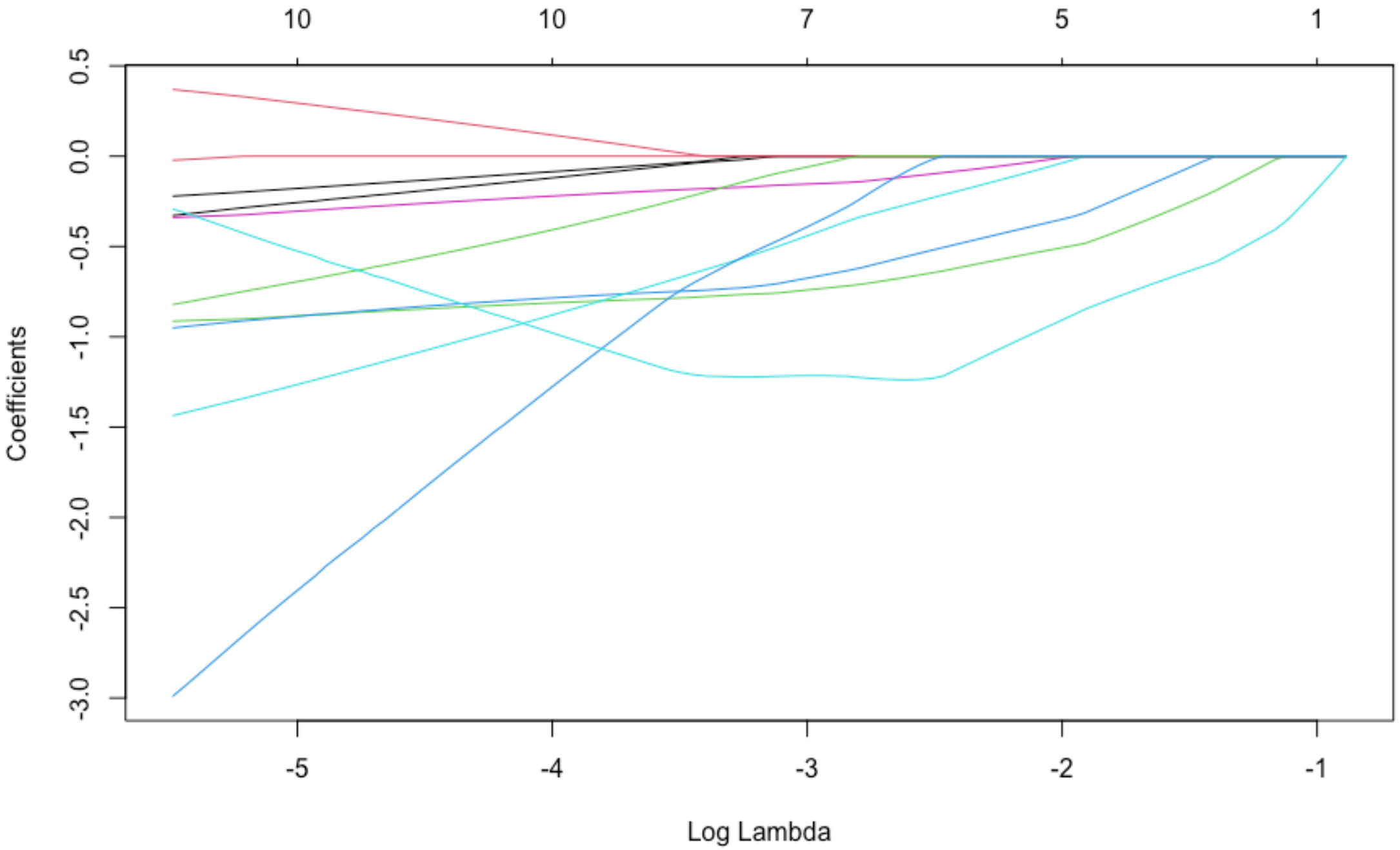
RIDGE OR LASSO?

- In general, the advantage of the lasso is that it performs an explicit variable selection, by estimating some coefficients exactly to zero.
- Ridge regression works better when the predictors are highly correlated, as is often the case in “omics” data.
- The two approaches can be compared using cross-validation procedures.

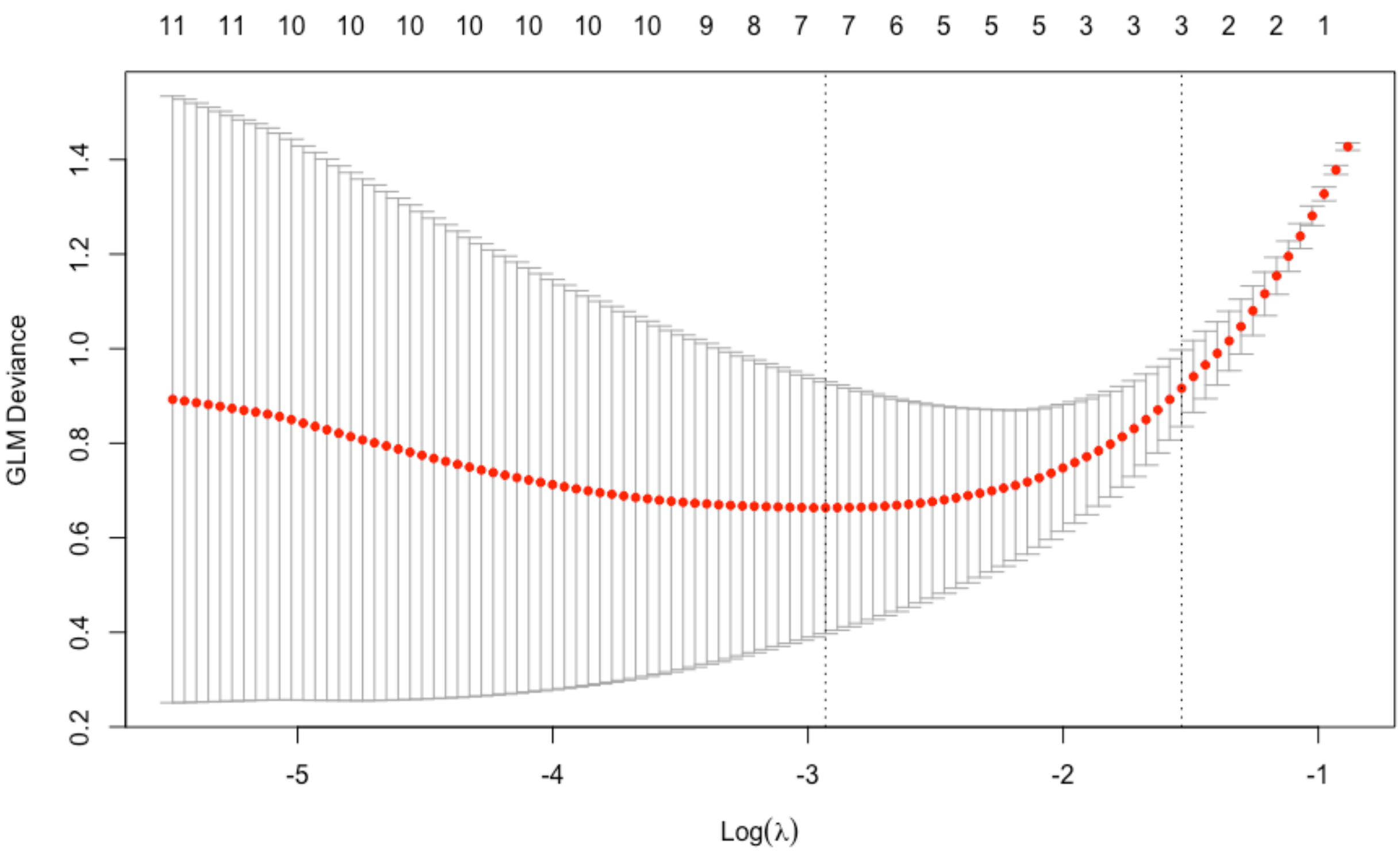
HOW MUCH PENALTY?

- One important choice is the penalty parameter, usually indicated by λ.
- Intuitively, the larger the value of λ the more the coefficients will be shrunk to zero.
- Usually, we let the data choose λ by trying a grid of values and comparing them in a cross-validation procedure.

EXAMPLE: FIEDLER DATA



EXAMPLE: FIEDLER DATA



EXAMPLE: FIEDLER DATA

	Estimate
(Intercept)	-0.03
x43	-0.35
x44	-0.17
X158	-0.72

FURTHER READING

- Modern statistics for modern biology by S. Holmes and W. Huber
- R for Data Science by H. Wickham and G. Grolemund
- Laurent Gatto has a lot of teaching material on how to analyze MS data with R/Bioconductor.
- Introduction to Statistical Learning by James, Witten, Hastie, and Tibshirani.