Workshop on "Enhancing your Computer Coding Skills"

Exploratory data analysis and clustering II

Thierry Denœux

tdenoeux@utc.fr
https://www.hds.utc.fr/~tdenoeux

Université de technologie de Compiègne

August 2022

Outline

1. Visualization of multidimensional data
   - Principal Component Analysis
   - Multidimensional Scaling
2. Dissimilarity measures for complex data
   - Qualitative and mixed-type data
   - Time series
3. Clustering dissimilarity data
   - Partitioning around medoids
   - Hierarchical Clustering

Overview

Motivation

- We have seen that 2D can be easily represented by scatter plots and 2D bar plots.
- This idea can be extended to p-dimensional data when p remains small (say, up to 5 or 6) using matrix plots (matrix representation of pairwise scatter plots).
- For higher values of p, visualizing the data becomes challenging.
- In this section, we will study two methods for mapping multidimensional data onto a smaller-dimension feature space amenable to visualization.
- These methods also allow one to find low-dimensional representations of of distance/dissimilarity data.

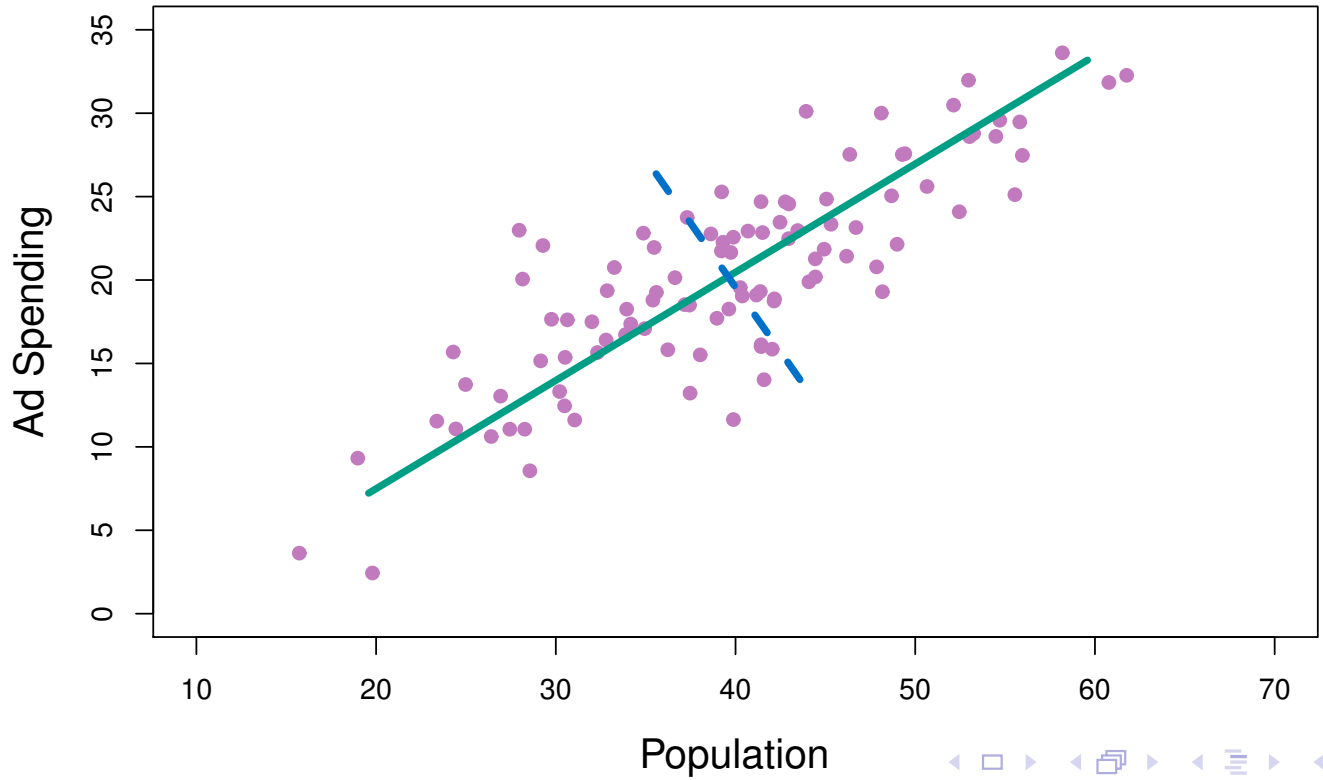
Overview

Basic idea

- We consider a cross-section $\mathcal{X} = \{x_1, \ldots, x_n\}$ of p variables (features, attributes) observed for n instances (observations, objects). We denote by x_{ij} the value of attribute j for object i.
- When $p \geq 3$ representing such data is difficult.
- One solution is to project dataset $\mathcal{X}$ onto a q-dimensional subspace, with $q < p$.
- By dropping $p - q$ dimensions, some information will be lost. The projection should be done in such a way that the loss of information is minimized.

Basic idea (continued)

Principal Component Analysis (PCA) finds orthogonal directions in input space along which the projected data have maximal variance. Each direction is defined by a vector u_k such that $\|u_k\| = 1$. The coordinate of observation i along the corresponding axis (called the principal component score) is $z_{ik} = u_k^T x_i$.



Scatter matrix

- The scatter of the data is described by the empirical covariance (scatter) matrix

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

  where $\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the sample mean.

- Let $X = (x_{ij})$ be the $n \times p$ data matrix, and assume that the data have been centered, so that $\overline{x} = 0$. We can write

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

Some properties of matrix S

- Matrix S is symmetric: $S^T = S$
- From now on, we assume that X has full column rank, i.e., no variable is a linear combination of other variables.
- Matrix S is then positive definite, i.e., for any nonzero vector u, $u^T S u > 0$.
- We recall that an eigenvector of matrix S is a nonzero vector u such that

$$\boxed{Su = \lambda u}$$

  for some real λ called the corresponding eigenvalue.
- As S is positive definite, its eigenvalues are all positive, and its eigenvectors are orthogonal.

First principal component

- Let u be a p-dimensional vector such that $\|u\| = u^T u = 1$, and $z_i = u^T x_i$ the coordinate of observation i along the axis directed by u. The vector $z = (z_1, \ldots, z_n)^T$ can be written as $z = Xu$.
- Vector u_1 such that the z_i's have maximum variance $\frac{1}{n}\sum_{i=1}^{n} z_i^2$ is

$$u_1 = \arg\max_{\|u\|=1} \sum_{i=1}^{n} (u^T x_i)^2 = \arg\max_{\|u\|=1} \|Xu\|^2$$
$$= \arg\max_{\|u\|=1} u^T \underbrace{X^T X}_{S} u$$

- It can be shown that u_1 is the eigenvector of matrix S with the largest eigenvalue λ_1.

Next principal components

- Having found u_1, we search for vector u that maximizes

$$\sum_{i=1}^{n}(u^T x_i)^2 = u^T S u$$

  subject to the constraints $\|u\| = 1$, $u^T u_1 = 0$.
- The solution is the eigenvector u_2 of S with the second largest eigenvalue λ_2.
- Continuing this line of reasoning, we obtain p vectors $u_1, \ldots, u_p$ with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p$.
- We can write $Z = XU$, where $U = (u_1, \ldots, u_p)$ is the $p \times p$ matrix (called the loading matrix) whose columns are the p eigenvectors. The columns of Z are p new variables called the principal components.

Explained variance

- The scatter matrix of Z is

$$(XU)^T XU = U^T \underbrace{X^T X}_{S} U = U^T \underbrace{SU}_{\Lambda U} = \Lambda \underbrace{U^T U}_{I_p},$$

  where $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_p)$ is the diagonal matrix containing the p eigenvalues of S.
- Consequently,

$$\text{tr}(\Lambda) = \text{tr}[U^T (SU)] = \text{tr}[(SU)U^T] = \text{tr}(S)$$

- Hence, the sum of the eigenvalues is equal to the total variance (the sum of the variances of the p original variables).
- The proportion of the variance explained by the first q components is

$$\sum_{j=1}^{q} \lambda_j \Big/ \sum_{j=1}^{p} \lambda_j$$

Overview

Main idea

- Multidimensional Scaling (MDS) is a set of techniques for representing distance or dissimilarity data.
- Given a $n \times n$ matrix $\Delta = (\delta_{ij})$ of dissimilarities, MDS finds a configuration of n points in a q-dimensional space, such that the (Euclidean) distances between points approximate the dissimilarities.
- We start by formally defining the notions of "distance" and "dissimilarity".

Distance

Definition (Distance)

A distance on a set X is a mapping $d : X^2 \to \mathbb{R}_+$ verifying the following properties

1. $d(x, y) = 0 \Leftrightarrow x = y$
2. $d(x, y) = d(y, x)$ *[symmetry]*
3. $d(x, z) \leq d(x, y) + d(y, z)$ *[triangular inequality]*

Examples of distances in $\mathbb{R}^p$:

- Euclidean: $d(x, y) = \sqrt{\sum_{j=1}^{p} (x_j - y_j)^2}$
- Manhattan: $d(x, y) = \sum_{j=1}^{p} |x_j - y_j|$
- Generalization: order-q Minkowski: $d(x, y) = \left(\sum_{j=1}^{p} (x_j - y_j)^q \right)^{1/q}$ $q \geq 1$

Dissimilarity

Definition (Dissimilarity)

A dissimilarity measure on a set X is a mapping $\delta : X^2 \to \mathbb{R}_+$ verifying the following properties

1. $\delta(x, x) = 0$
2. $\delta(x, y) = \delta(y, x)$ *[symmetry]*

Examples:

- Travel times between cities
- Subjectively assessed dissimilarities between products, objects, etc.
- More later...

Stress function

- Let $\Delta = (\delta_{ij})$ be a dissimilarity, and let X be an $n \times p$ matrix describing a configuration of n points in $\mathbb{R}^p$. We denote by $d_{ij}(X)$ the Euclidean distance between points x_i and x_j.
- MDS consists in finding X minimizing discrepancies between the distances $d_{ij}(X)$ and the dissimilarities δ_{ij}. Discrepancies are measured by a stress (error) function.
- Simplest method: we minimize the raw stress defined as

$$\text{Stress}(X) = \sum_{i<j} (\delta_{ij} - d_{ij}(X))^2$$

Other stress functions

Depending on the scale in which dissimilarities are expressed, we can apply different monotonic transformations to the δ_{ij}'s in the stress function:

Ratio MDS

$$\text{Stress}(X, a) = \sum_{i<j} (a\delta_{ij} - d_{ij}(X))^2$$

Interval MDS

$$\text{Stress}(X, a, b) = \sum_{i<j} (a\delta_{ij} + b - d_{ij}(X))^2$$

Ordinal (nonmetric) MDS

$$\text{Stress}(X, f) = \sum_{i<j} (f(\delta_{ij}) - d_{ij}(X))^2$$

where f is a monotone function.

Nonmetric MDS is particularly recommended when dissimilarities are measured on an ordinal scale (e.g., "small", "medium", "large").

Overview

Motivation

- Until now, we have studied clustering algorithms applicable to cross-sectional data with numerical attributes.
- In this section we will review some dissimilarity measures for more complex data, namely: cross-sectional data with qualitative or mixed (qualitative/numerical) attributes and time series.
- Having defined a dissimilarity measure for some kind of data, there are, at least, two strategies for clustering:
  1. Describe the data in a space of numerical attributes using MDS, and use clustering techniques for numerical attribute data such as HCM or FCM
  2. Use a clustering algorithm that directly uses dissimilarities as inputs; such algorithms will be studied later (PAM, hierarchical clustering)

Overview

Binary variables

- Let us assume that we have p binary variables, i.e., variables with two possible outcomes, encoded as 0 or 1.
- We can distinguish between two cases:

  Symmetric case: the two outcomes are equally important, the encoding as 0 or 1 is arbitrary (e.g., sex, right-handed); two individuals having both 0 or 1 values are equally similar;

  Asymmetric case: both outcomes are not equally important, for instance, they correspond to the presence or absence of a rare attribute. Not having this attribute (e.g., being a Nobel prize winner) does not make two individuals similar.

Dissimilarity measures for binary data

- Given n objects and p binary attributes, we can compute the following contingency table for any two objects i and j:

		Object j	
		0	1
object i	0	a	b
	1	c	d

with $p = a + b + c + d$.

- In the case of symmetric attributes, the dissimilarity between objects i and j can be measured by the simple matching coefficient

$$\delta_{ij} = \frac{b + c}{a + b + c + d}$$

- In the case of asymmetric attributes, we often use the Jaccard index

$$\delta_{ij} = \frac{b + c}{b + c + d}$$

Nominal variables

- A nominal (or categorical) variable has M unordered outcomes (e.g., marital status with outcomes bachelor/married/divorced/widowed)
- In the case of p nominal variables, the most common dissimilarity measure is the simple matching coefficient:

$$\boxed{\delta_{ij} = \frac{p - q}{p}}$$

where q is the number of matches (number of variables that have the same outcomes for objects i and j).

Ordinal variables

- An ordinal variable is a variable with a finite number of ordered outcomes, for instance, very low'/low/medium/high/very high
- Treating such a variable as a nominal one and ignoring the order would result in a loss of information.
- Common approach: assign to each outcome its rank in $\{1, \ldots, M\}$ and convert the ranks to the 0-1 range. The rank r_{ik} of object i for variable k is replaced by

$$\boxed{z_{ik} = \frac{r_{ik} - 1}{M_k - 1}}$$

  where M_k is the number of outcomes of variable k.
- The transformed ranks z_{ik} are then treated as numerical variables, with distances measured by the Euclidean or Manhattan distance.

Mixed-type data

- Mixed-type data are data with attributes of different types.
- For this kind of data, Gower (1971) proposed to compute the dissimilarity between two objects i and j as

$$\delta_{ij}^{(k)} = \frac{\sum_{k=1}^{p} \gamma_{ij}^{(k)} \delta_{ij}^{(k)}}{\sum_{k=1}^{p} \gamma_{ij}^{(k)}}$$

  where $\gamma_{ij}^{(k)} = 1$ if both measurements x_{ik} and x_{jk} for the variable k are nonmissing, and $\gamma_{ij}^{(k)} = 0$ otherwise; moreover, $\gamma_{ij}^{(k)}$ is also put equal to 0 when variable k is an asymmetric binary attribute and objects i and j constitute a 0-0 match.
- If k is a numerical attribute, then

$$\delta_{ij}^{(k)} = \frac{|x_{ik} - x_{jk}|}{R_k}$$

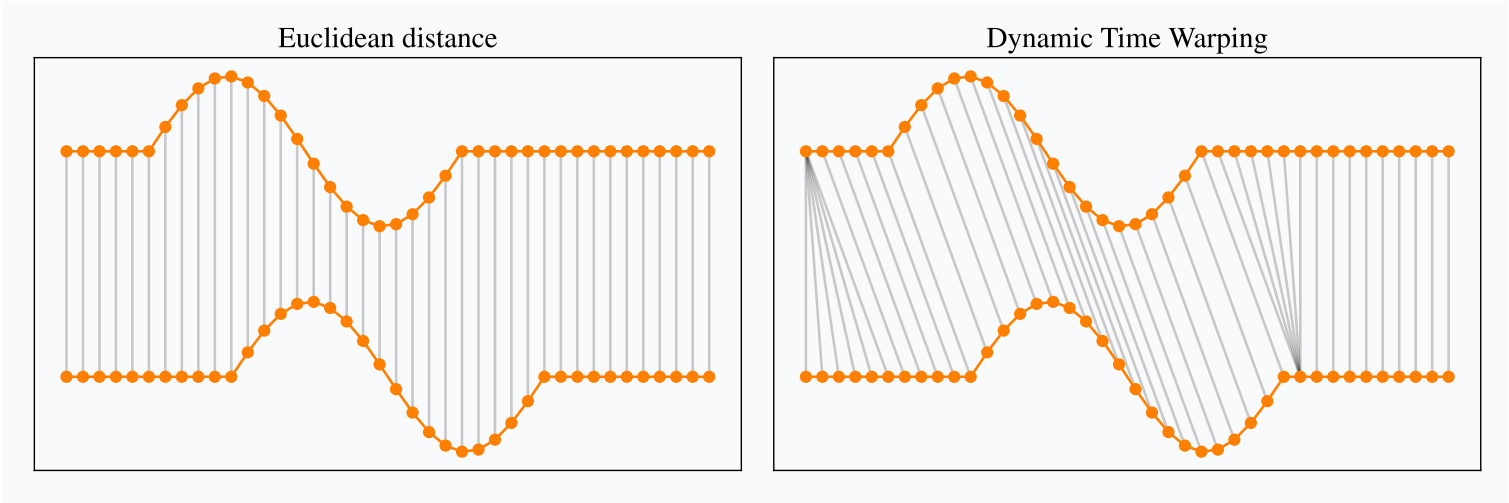
  where R_k is the range of variable k.

Overview

Measuring the dissimilarity between time series

- By defining a dissimilarity measure between time series, we will be able to apply clustering and multidimensional data visualization techniques to times series data.
- There are several ways to measure the dissimilarity between time series or sequences, depending on what we want to achieve.
- Often, two sequences (possibly of different lengths) are considered to be similar if they show the same pattern, which may occur at different times and at different time scales.

Dynamic time warping



- Treating the sequences as vectors and computing, e.g., the Euclidean distance between vectors will not capture the similarity between patterns. Furthermore, this is not possible when the sequences may have different lengths.
- Dynamic time warping (DTW) is a widely used technique to compute the similarity of sequences by finding the best alignment between the sequences, i.e., the best matching relation between the indices of the two sequences.

Problem formulation

- The optimization problem writes

$$DTW(x, x') = \min_{\pi \in \mathcal{A}(x,x')} \sum_{(i,j)\in\pi} d(x_i, x'_j)$$

where $d(x_i, x'_j)$ is a distance between x_i and x'_j (which may be vectors), and π is an alignment path defined as a sequence of K index pairs $((i_0, j_0), \ldots, (i_{K-1}, j_{K-1}))$ such that

  - The first index from the first sequence is matched with the first index from the other sequence: $(i_0, j_0) = (0, 0)$
  - The last index from the first sequence is matched with the last index from the other sequence: $(i_{K-1}; j_{K-1}) = (n-1, m-1)$
  - The sequence is monotonically increasing in both i and j and all time series indices appear at least once:

$$i_{k-1} \leq i_k \leq i_{k-1} + 1 \quad \text{and} \quad j_{k-1} \leq j_k \leq j_{k-1} + 1$$

Solution

- Although the search space is finite, exhaustive search becomes quickly intractable even for moderate time series lengths.
- The problem is considerably simplified by existence of the following recurrence equation:

$$R_{i,j} = d(x_i, x_j) + \min(R_{i-1,j}, R_{i,j_1}, R_{i-1,j-1})$$

  where R_{ij} is the value of the cost function up to timestamps i and j.
- A dynamic programming algorithm that exploits this equation can find an exact solution in $O(nm)$ time.

Overview

Motivation

- We have seen that a dissimilarity matrix can be clustered by first computing a corresponding configuration of points in $\mathbb{R}^p$ using MDS.
- However, the fit between dissimilarities and distances between points in the configuration is not perfect: some information is lost in the process.
- As an alternative, we can seek to cluster the data directly using the dissimilarity matrix itself.
- There exist many clustering algorithms for dissimilarity data. Here, we will see two of the most widely used approaches:
  1. The Partitioning Around Medoids (PAM) algorithm
  2. Hierarchical clustering

Overview

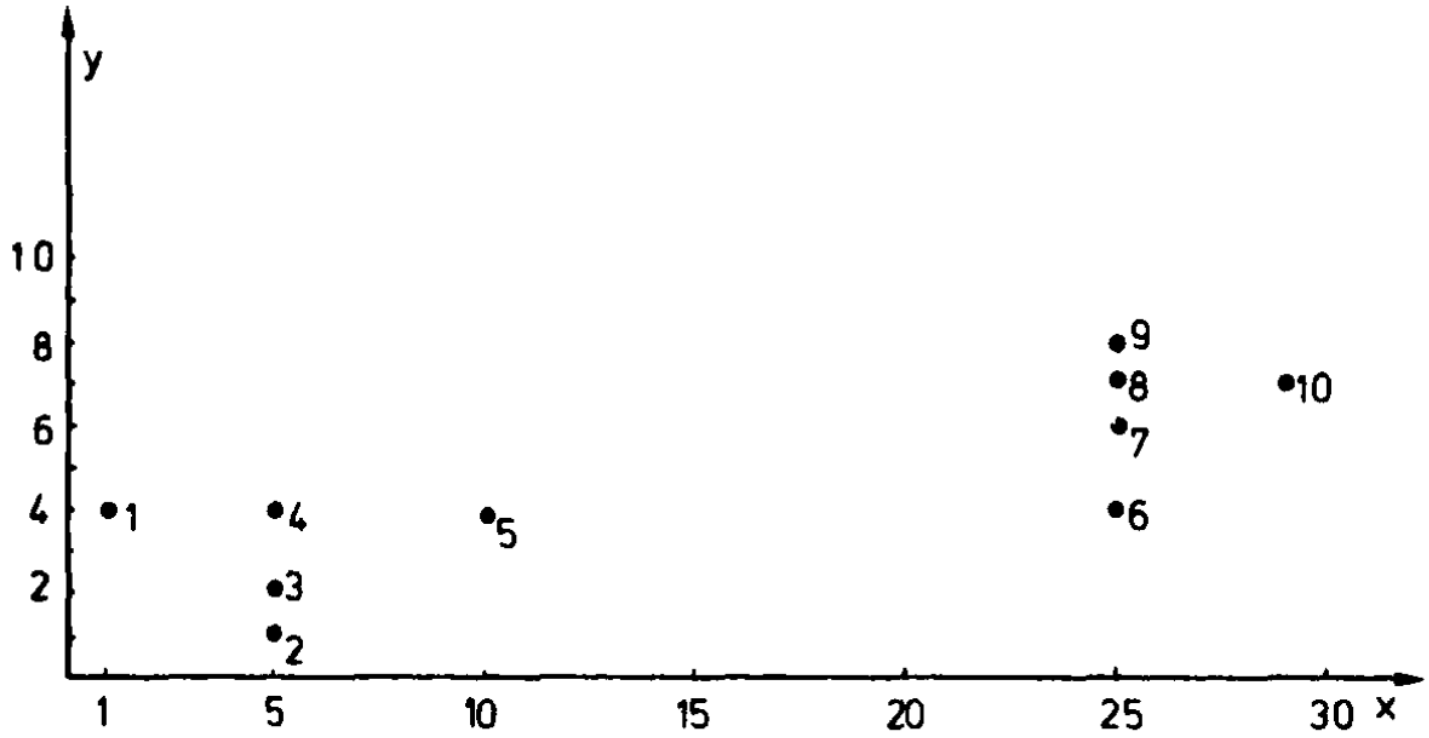
Main idea

- The PAM algorithm is similar to HCM in that it alternates between two phases:
  - Describing each class by a prototype
  - Assigning each object to the nearest prototype
- However, in PAM, prototypes are chosen among the objects themselves (and not as means of attribute vectors in each class as in HCM). Such prototypes are called medoids.
- As a consequence, the algorithm only needs a distance or dissimilarity matrix, and not a matrix of attributes.

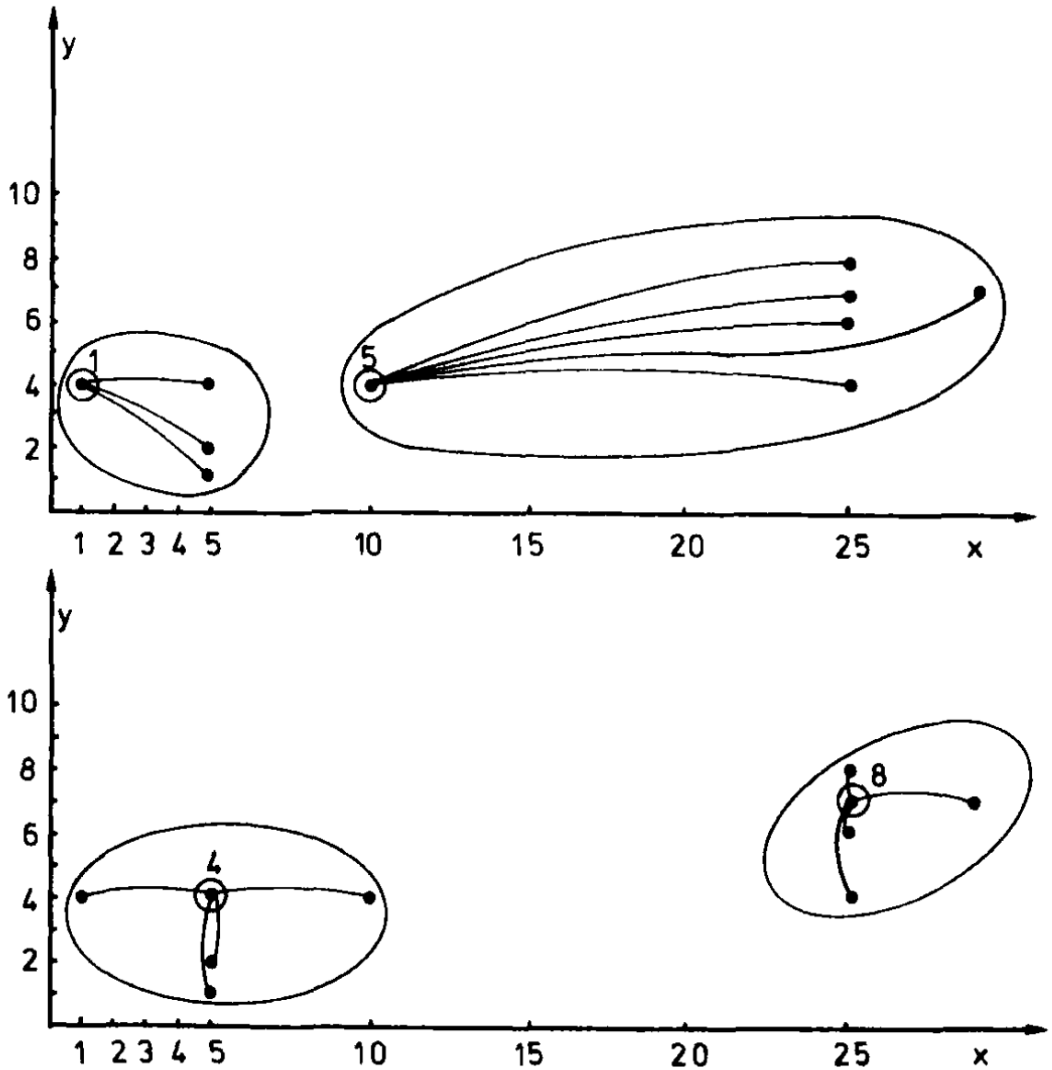
Description of the algorithm

1. Initialization: Choose c objects to become the medoids
2. Build phase: Assign every entity to its closest medoid
3. Swap phase:
   - For each cluster, search if any of the entities of the cluster lowers the average dissimilarity coefficient; if it does, select the entity that lowers this coefficient the most as the medoid for this cluster
   - If at least one medoid has changed go to (2), else end the algorithm

Example



Example (continued)



Discussion

- The PAM algorithm is more robust than HCM to outliers.
- However, his robustness comes at the expense of more computations: the time complexity of PAM is roughly $O(c(n-c)^2)$ (against $O(nc)$ for HCM).
- Consequently, PAM is limited to datasets of small/medium size.
- There exist variants for larger datasets.

Overview

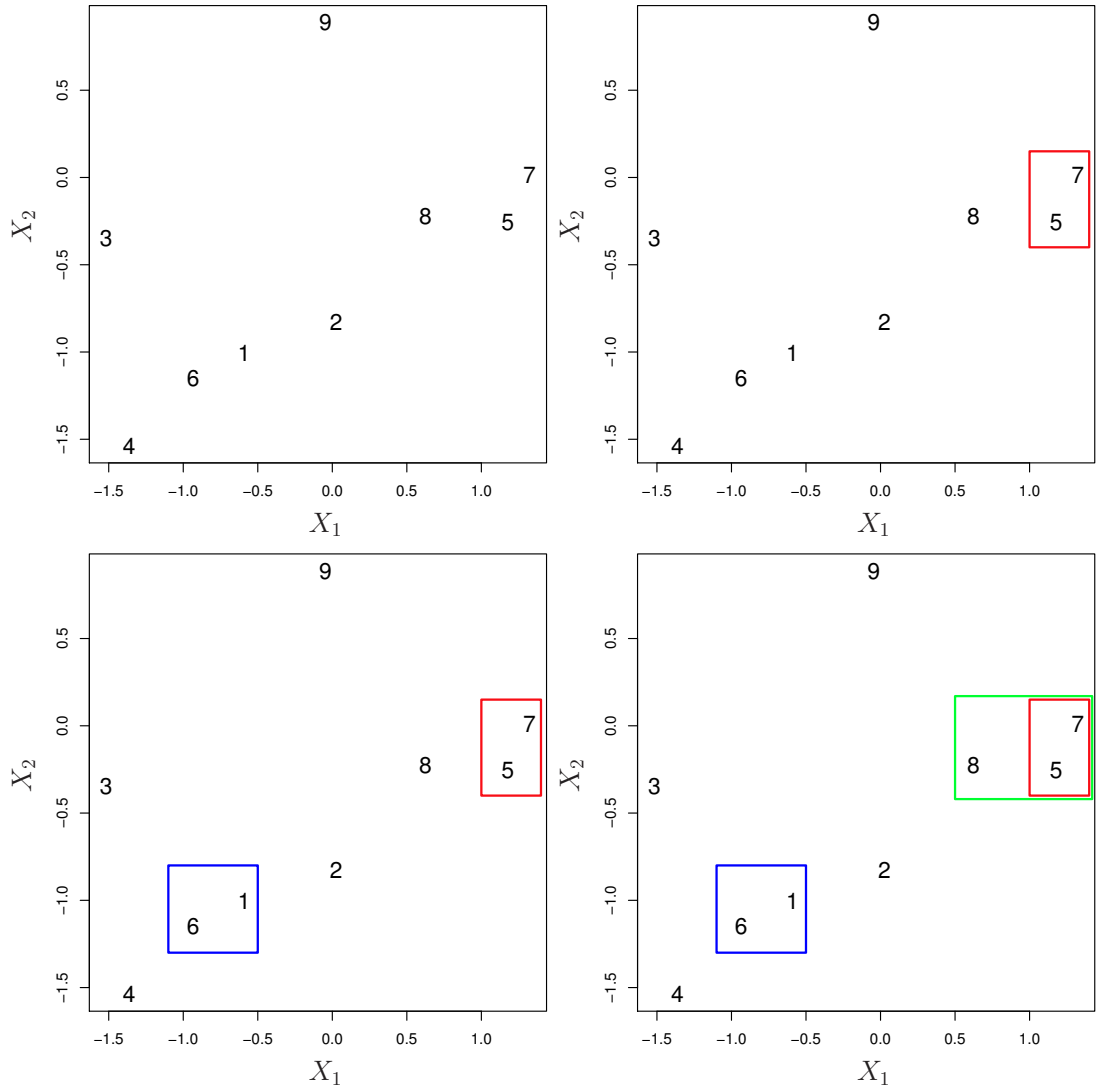
Notion of hierarchy

- We have seen that determining the "best" number of clusters is a difficult problem in partitional and fuzzy clustering.
- As an alternative, we can build a sequence of partitions $P_n, P_{n-1}, \ldots, P_2, P_1$, such that
  - Each partition P_k has exactly k clusters
  - Each partition P_k is obtained by merging 2 clusters in partition P_{k+1}
- Such a sequence of partitions is called a hierarchy.

Hierarchical clustering

- Hierarchical clustering methods do not require the user to supply a number of clusters, but only a way to compute the dissimilarity between two clusters, based on dissimilarities among the observations in the two clusters.
- They produce hierarchical representations in which the clusters at each level of the hierarchy are created by merging clusters at the next lower level.
- At the lowest level, each cluster contains a single observation. At the highest level there is only one cluster containing all of the data.

Example



Principle

- Agglomerative clustering algorithms begin with every observation representing a singleton cluster.
- At each of the $n - 1$ steps the closest (most similar) two clusters are merged into a single cluster, producing one less cluster at the next higher level.
- Therefore, a measure of dissimilarity between two clusters (groups of observations) must be defined.

Dissimilarity between clusters

- Let G and H represent two groups. The dissimilarity $d(G, H)$ between G and H is computed from the set of pairwise observation dissimilarities $\{d_{ij} : i \in G, j \in H\}$.
- Three main definitions:

  Single linkage

  $$d(G, H) = \min_{i \in G, j \in H} d_{ij}$$

  Complete linkage

  $$d(G, H) = \max_{i \in G, j \in H} d_{ij}$$

  Average linkage

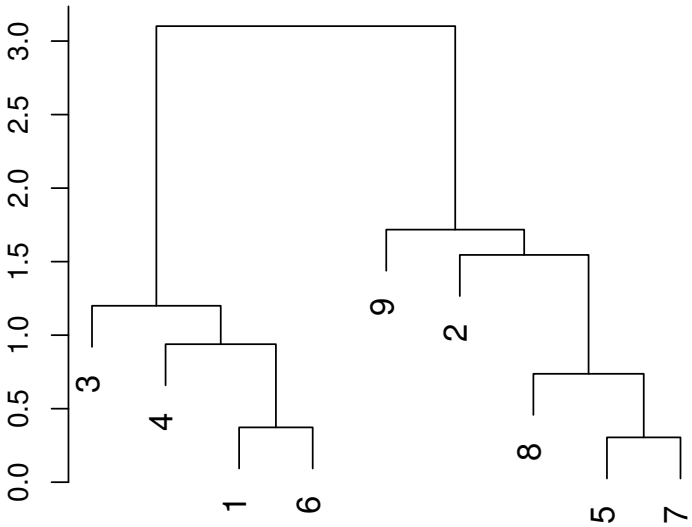
  $$d(G, H) = \frac{1}{|G||H|} \sum_{i \in G, j \in H} d_{ij}$$

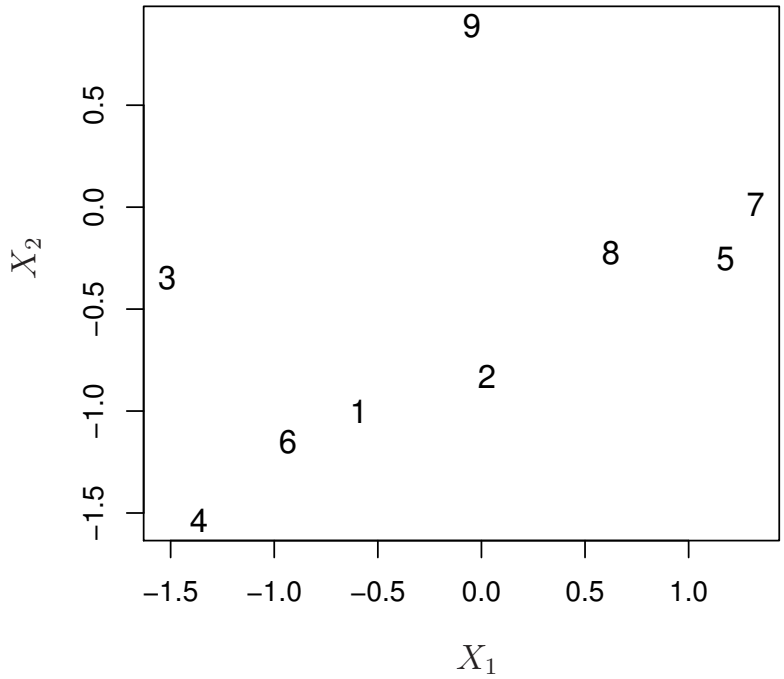
Criteria for selecting a distance measure

- If the data dissimilarities d_{ij} exhibit a strong clustering tendency, with each of the clusters being compact and well-separated from others, then all three methods produce similar results. To the extent this is not the case, results will differ.
- Single linkage tends to created elongated clusters due to the chaining effect (observations linked by a series of close intermediate observations).
- In contrast, complete linkage tends to produce compact and small clusters.
- Group average clustering represents a compromise between the two extremes of single and complete linkage: it attempts to produce relatively compact clusters that are relatively far apart.

Dendrogram

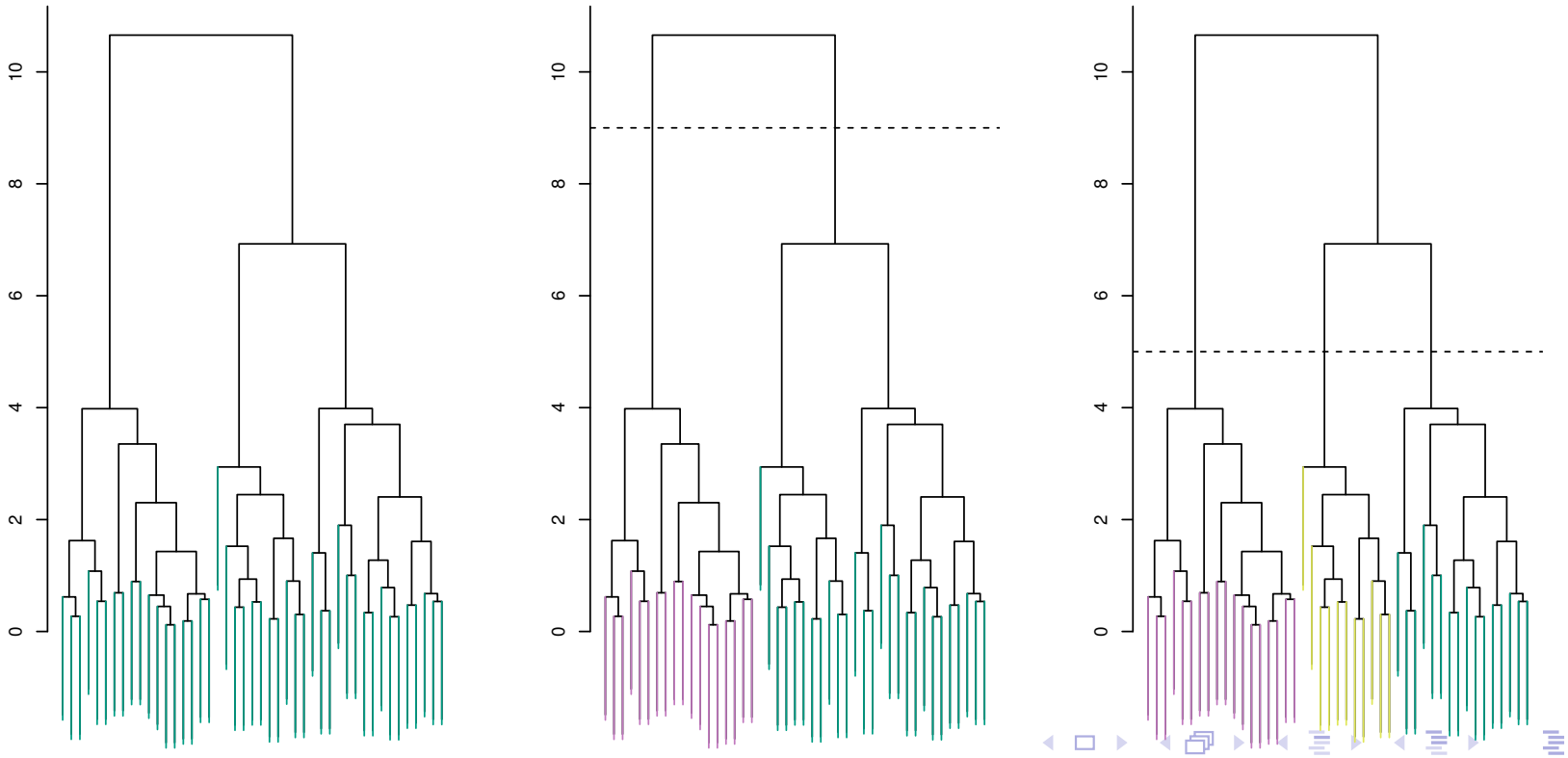
The hierarchy can be plotted as a tree such that the height of each node is proportional to the value of the between-group dissimilarity between its two sons. The terminal nodes representing individual observations are all plotted at zero height. This is called a dendrogram.





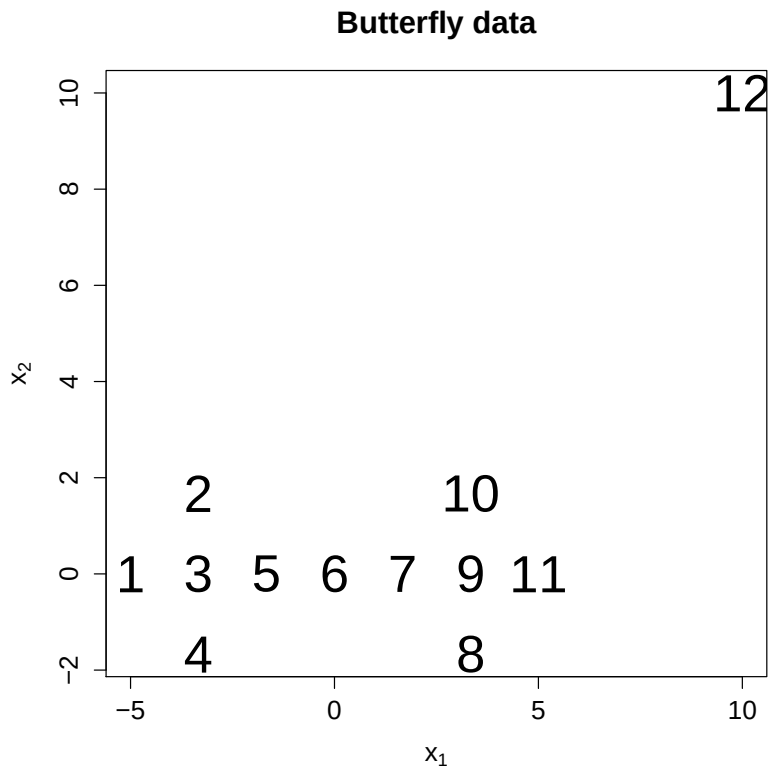
Cutting the dendrogram

Cutting the dendrogram horizontally at a particular height partitions the data into disjoint clusters. Groups that merge at high values, relative to the merger values of the subgroups contained within them lower in the tree, are candidates for natural clusters.



Example in R: Butterfly data

```
D<-dist(x)
h<-hclust(D,method="complete")
plot(h,main="Complete Link")
```



Result