STA 141A – Fundamentals of Statistical Data Science

Department of Statistics; University of California, Davis

Instructor: Dr. Akira Horiguchi (ahoriguchi@ucdavis.edu)
A01 TA: Zhentao Li (ztlli@ucdavis.edu)
A02 TA: Zijie Tian (zijtian@ucdavis.edu)
A03 TA: Lingyou Pang (lyopang@ucdavis.edu)

Section 9: Unsupervised learning

Spring 2025 (Mar 31 – Jun 05), MWF, 01:10 PM – 02:00 PM, Young 198

Comments

In supervised learning, we have access to some predictors $X_1, \ldots, X_p$ measured on n observations, and a response Y.

In unsupervised learning, we have predictors $X_1, \ldots, X_p$ but no response.

- Without a response, we can still do analysis given the predictors $X_1, \ldots, X_p$, and analyze their association to each other.
- Often performed as part of an exploratory data analysis.
- It can be hard to assess the results obtained from unsupervised learning methods; there is no "truth" to compare to. (In contrast, in supervised learning the "truth" is the response Y.)

Unsupervised learning tasks: *clustering* and *dimension reduction*.

Overview

Based on Chapter 12 of ISL book James et al. (2021).

- For more R code examples, see R Markdown files in `https://www.statlearning.com/resources-second-edition`

Section 9: Unsupervised learning

- *K*-means clustering
- Hierarchical clustering
- Principal component analysis (PCA)

Unsupervised learning

K-means clustering

Task: find homogeneous subgroups (i.e., *clusters*) among observations.

- "Market segmentation" aims to identify subgroups of people who might be more receptive to certain kind of advertisements/products etc.
- Flow cytometry.

Figure 1: From James et al. (2021).

Two common techniques: *K-means clustering* and *hierarchical clustering*.

Idea of K-means clustering

K-means clustering partitions observations into K non-overlapping clusters.

- The user chooses the value of K before performing K-means clustering.
- *Clusters* are sets $C_1, \ldots, C_K$ with the following features:
  - $C_1 \cup C_2 \cup \cdots \cup C_K = \{1, \ldots, n\}$ (each obsn belongs to at least one cluster);
  - $C_k \cap C_l = \emptyset$ for all $k \neq l$ (no observation belongs to more than one cluster).
- "Good" clustering: if the obsns in each cluster are close to each other, i.e., if the *within-cluster variation* is relatively small.
- Several ways to define *within-cluster variation*. The most common choice involves the squared euclidian distance (common distance of vectors).
- For obsns $x_1, \ldots, x_n \in \mathbb{R}$, within-cluster variation of a cluster C defined by

$$W(C) := \frac{1}{\#C} \sum_{i,i' \in C} (x_i - x_{i'})^2 . \tag{1}$$

- For obsns $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$, within-cluster variation of a cluster C defined by

$$W(C) := \frac{1}{\#C} \sum_{i,i' \in C} \|\mathbf{x}_i - \mathbf{x}_{i'}\|_2^2 = \frac{1}{\#C} \sum_{i,i' \in C} \sum_{j=1}^{p} \left(x_{ij} - x_{i'j}\right)^2 . \tag{2}$$

- We want to find clusters $C_1, \ldots, C_K$ that minimize $\sum_{k=1}^{K} W(C_k)$.

Comments:

- The above minimization problem is very difficult to solve precisely, since there are almost K^n ways to partition n observations into K clusters.
- The following algorithm can be shown to provide a local optimum.

Algorithm:

1. Randomly assign a number from 1 to K (K is pre-defined) to each obsn.
2. Iterate steps (a) and (b) until the cluster assignments stop changing:
   (a) For each cluster, compute *cluster centroid* (mean of all obsns in the cluster).
   (b) Assign each observation to the cluster whose centroid is the closest.

Comments:

- K-means clustering derives its name from the fact that the cluster centroids are computed as the mean of each cluster's observations.
- Step 2 can be shown to never increase $\sum_{k=1}^{K} W(C_k)$ — will reduce it until at local minimum.
- Value of obtained local minimum will depend on initial (random) cluster assignment in Step 1.
- One should run the algorithm multiple times, and then choose best result.

Example: compute the centroid of a cluster that consists of 2-dim obsns

$(1, 2), (2, 1), (3, 2), (1, 0)$.

Simulation of K-means clustering

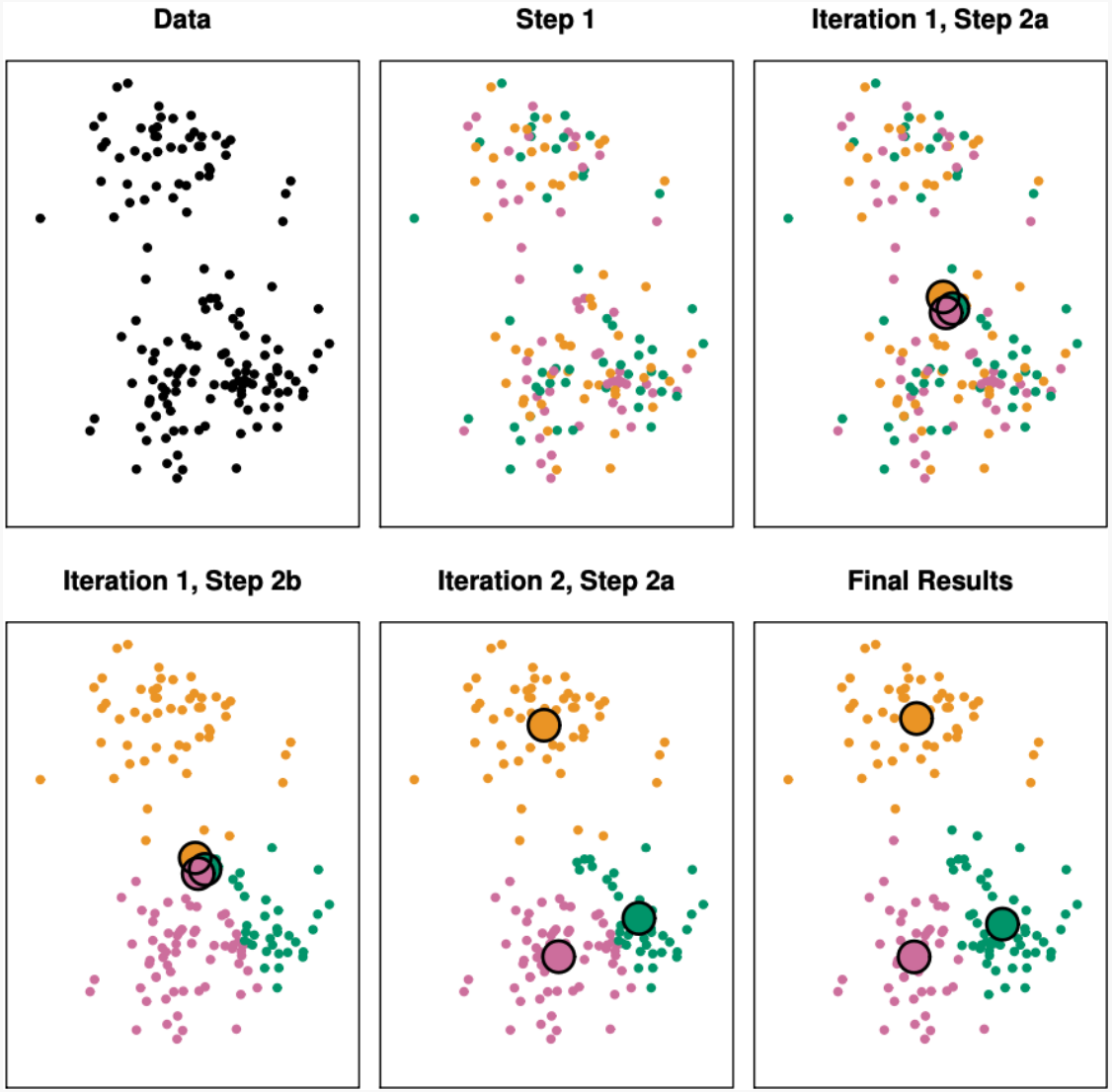

Figure 2: From James et al. (2021). 3-means clustering and 10 iterations.

Unsupervised learning

Hierarchical clustering

Motivation

- A potential disadvantage of K-means clustering is that it requires to prespecify number of clusters K.
- *Hierarchical clustering* is an alternative to K-means clustering, where no pre-defined number of clusters K is needed.
- In hierarchical clustering, we also have the advantage of having structured, tree-based illustration, called *dendrogram*.
- Here, we describe hierarchical clustering in sense of *bottom-up* clustering, meaning that the dendogram is built from the leaves "bottom up".

Interpreting a dendrogram

Suppose simulated data set is observed without class labels.

- Hierarchical clustering yields dendrogram. How to interpret?
- Each leaf of dendrogram represents one of the 45 obsns.
- As we move up tree, some leaves begin to fuse into branches. These correspond to obsns similar to each other. Then branches begin to fuse.
- For any two obsns, height of fusion indicates how different the obsns are. (Ignore horizontal proximity.)
- To identify clusters, make a horizontal cut across dendrogram.
- Height of cut controls number of clusters obtained.
- A single dendrogram can be used to get any number of clusters. Eyeball.

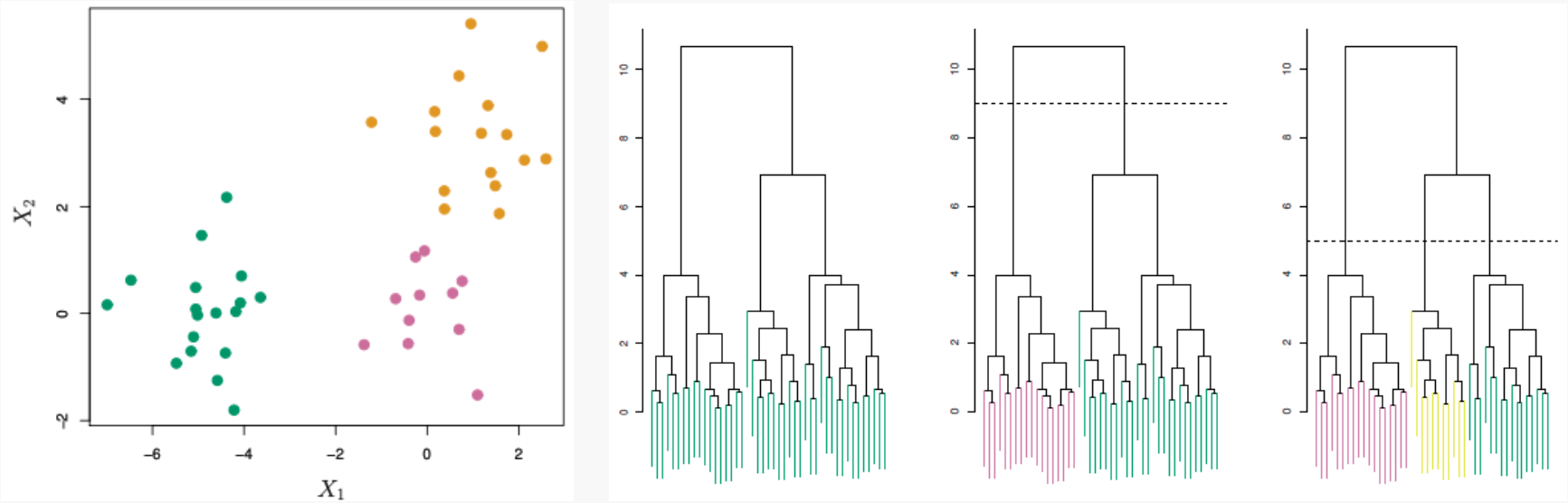

Figure 3: From James et al. (2021). 45 observations.

Closer look at a dendrogram

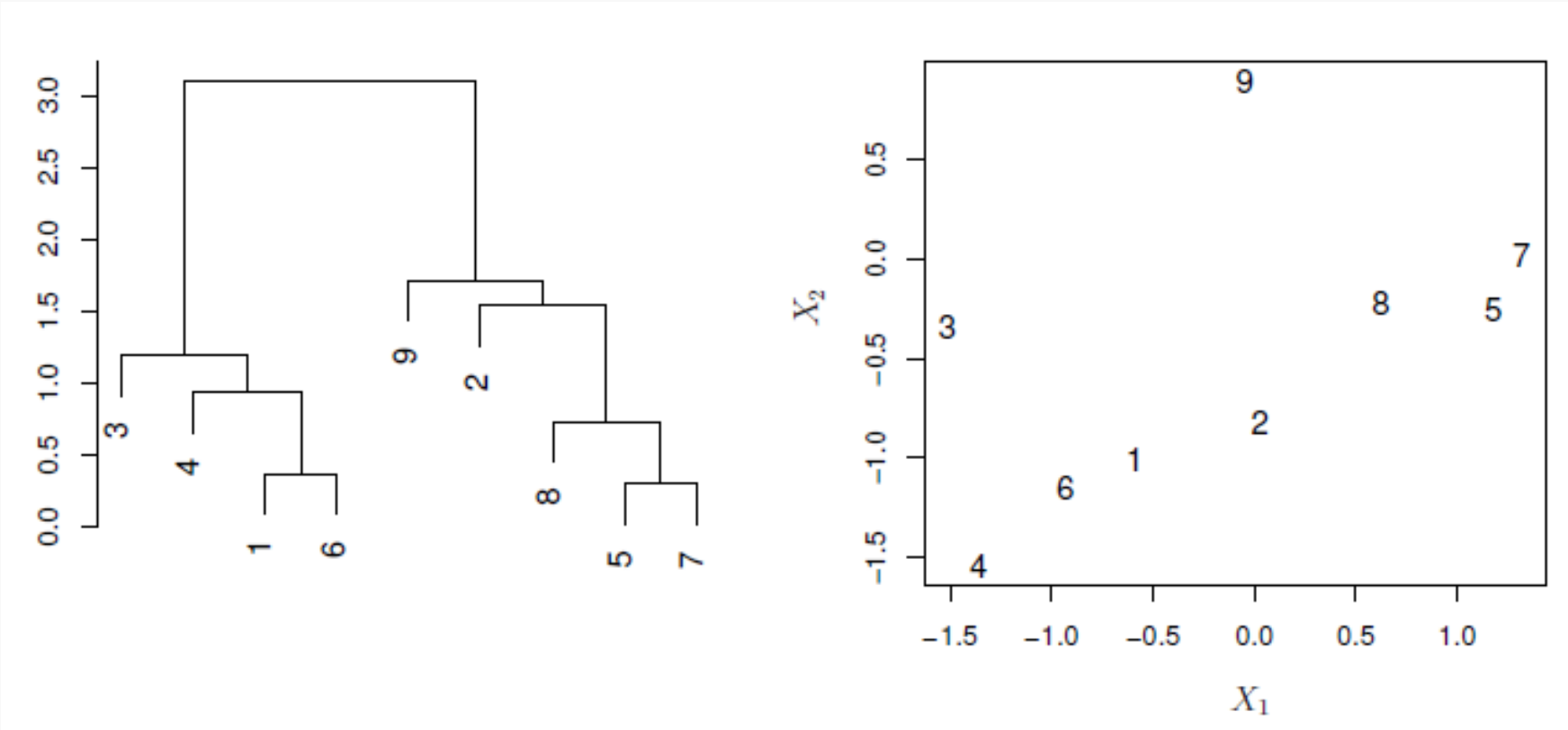

Figure 4: Figure by James et al. (2021). Left: A dendrogram generated using Euclidean distance and complete linkage. Right: The raw data used to generate the dendrogram.

Hierarchical clustering can sometimes lead to worse results than with K-means clustering.

- This is because if we have observations with two different features (e.g. "mother tongue") and also three other different features (e.g. "place of residence"), then K-means clustering with $K = 2$ would possibly split the observations by mother tongue, and with $K = 3$ by place of residence, which cannot realized by hierarchical clustering (as the clusters would be completely different and not nested).

Algorithm:

1. Begin with n obsns and a dissimilarity measure (often Euclidian distance), and treat each observation as a cluster.
2. For $i = n, n-1, \ldots, 2$:
   (a) Compute the $\binom{i}{2}$ pairwise inter-cluster dissimilarities among the i clusters.
   (b) Identify the pair of clusters that are least dissimilar (i.e. most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.

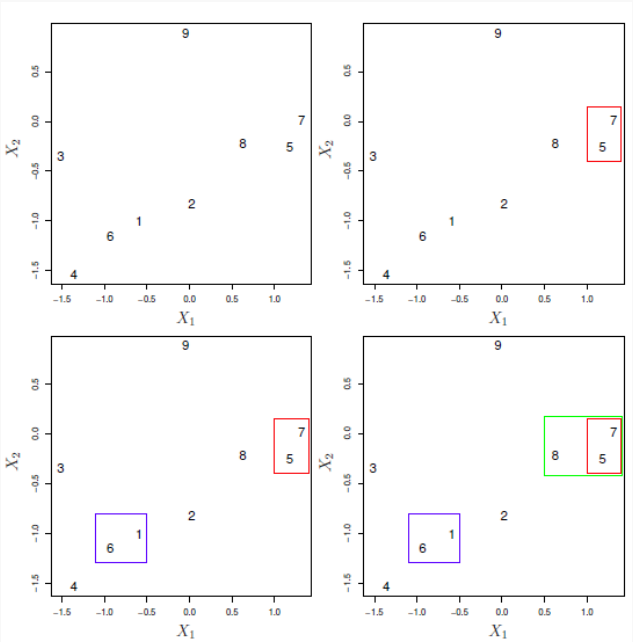

Figure 5: From James et al. (2021). First few steps of hierarchical clustering algorithm with complete linkage and Euclidean dist. Top Left to Top Right to Bottom Left to Bottom Right.

Algorithm comments

How to define dissimilarity between e.g., cluster $\{5, 7\}$ and cluster $\{8\}$?

- Need to extend dissimilarity to two groups of observations.
- *Linkages* define the dissimilarity between two groups of observations.
  1. *Complete*: computes all dissimilarities between an obsn in cluster A and an obsn in cluster B, and record largest of these $n_A n_B$ dissimiliarities.
  2. *Single*: same, except record smallest of these $n_A n_B$ dissimiliarities.
  3. *Average*: same, except record mean of these $n_A n_B$ dissimiliarities.
  4. *Centroid*: dissimilarity between two cluster centroids.

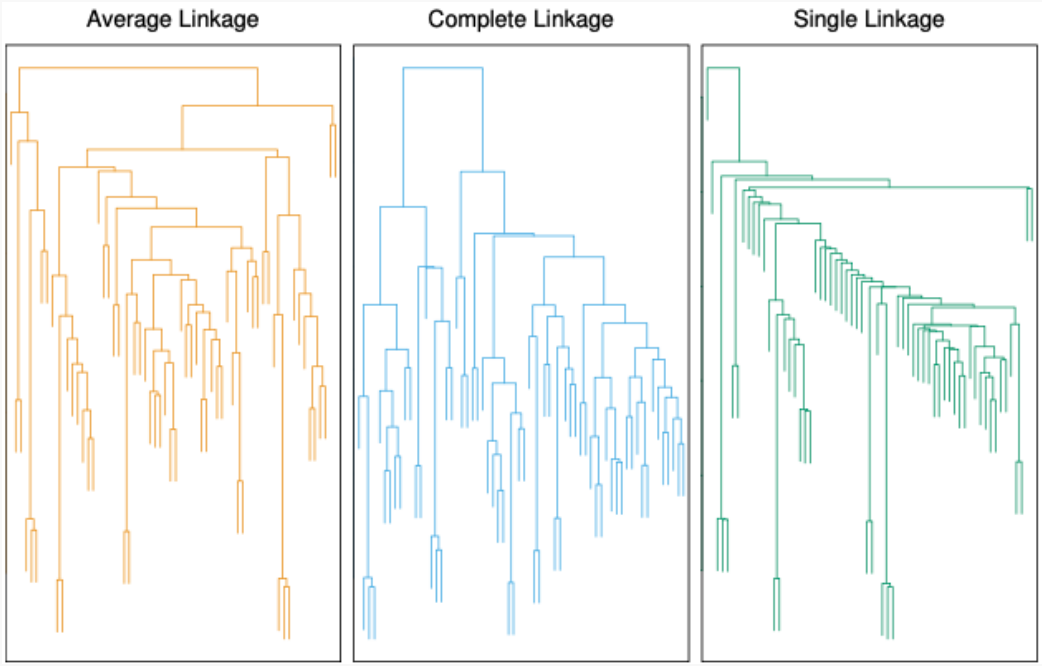

Figure 6: From James et al. (2021). Average, complete, and single linkage applied to an example data set. Average and complete linkage tend to yield more balanced clusters.

Issues in clustering

- Should observations first be standardized in some way? E.g. should variables be scaled to have standard deviation one?
- Hierarchical clustering:
  - What dissimilarity measure should be used?
  - What type of linkage should be used?
  - Where shall the dendogram be cut (i.e., how many clusters do we need/want)?
- *K*-means clustering: how many clusters should we look for?

Unsupervised learning

Principal component analysis (PCA)

Dimension reduction

Suppose we have n obsns on a set of p features $X_1, X_2, \ldots, X_p$.

- That is, suppose we have n data points $(x_{11}, x_{12}, \ldots, x_{1p})^\top, (x_{21}, x_{22}, \ldots, x_{2p})^\top, \ldots, (x_{n1}, x_{n2}, \ldots, x_{np})^\top$.
- Dimension reduction: reduce dimensionality of data while retaining as much information about the data as possible.
- Idea: not all p dimensions are equally interesting. E.g., if jth feature has almost the same value for all n obsns, do we really need to keep track of jth feature?
- Dimension reduction is useful for e.g.,
  - 2-dim scatterplots of data if $p = 10$.
  - image compression.
  - denoising images.

PCA – Idea

Principal components analysis (PCA) seeks a small number of dimensions that are as interesting as possible.

- Here "interesting" is measured by how much the n observations vary along the dimension.
- Each dimension (i.e., *principal component*) found by PCA is a *linear combination* of the p features.
  - A linear combination of the p features is defined as

    $$\phi_1 X_1 + \phi_2 X_2 + \cdots + \phi_p X_p$$

    for some *coefficient* values $\phi_1, \phi_2, \ldots, \phi_p$.
  - Given $X_1, X_2, \ldots, X_p$, a linear combination can be represented by the vector $\phi = (\phi_1, \phi_2, \ldots, \phi_p)$.
  - A linear combination is called *normalized* if $\|\phi\|_2^2 = 1$. (unit vector)
- The coefficients $\phi_1, \phi_2, \ldots, \phi_p$ of a principal component are called its *loadings*; the vector ϕ is called its *loading vector*.
- Principal component loading vectors are always orthogonal to each other.
  - Vectors $(a_1, \ldots, a_p)$ and $(b_1, \ldots, b_p)$ are called *orthogonal* or *perpendicular* to each other if $\sum_{j=1}^{p} a_j b_j = 0$ (i.e., if dot product is zero).
- PCA computes p orthogonal principal component loading vectors in order from "most interesting" to "least interesting".

First principal component

Given an $n \times p$ data matrix $\mathbf{X}$, how to compute the *first principal component*?

- Because we are only interested in variance, assume each column in $\mathbf{X}$ has mean zero.
- A linear combination $\phi_{11}, \phi_{21}, \ldots, \phi_{p1}$ leads to the n values

$$z_{i1} = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \cdots \phi_{n1} x_{in} \qquad i = 1, \ldots, n. \tag{3}$$

- As we assumed all data points to be centered, all z_{i1} are also centered, why (4) is just maximing the sample variance of all values $z_{11}, z_{21}, \ldots, z_{n1}$.
- First principal component loading vector is defined to be the normalized linear combination that maximizes

$$\frac{1}{n} \sum_{i=1}^{n} z_{i1}^2 . \tag{4}$$

- (Normalization ensures the variance is not arbitrarily large.)
- Because each column of $\mathbf{X}$ is centered, mean of $z_{11}, z_{21}, \ldots, z_{n1}$ is also zero. So maximizing (4) is same as maximizing sample variance of $z_{11}, z_{21}, \ldots, z_{n1}$.
- (4) can be maximized via *eigen decomposition* or *singular value decomposition* (see lab, but don't need to know this for exam).

First principal component

- Geometric interpretation: the first principal component loading vector defines a direction in feature space along which the data vary the most.
- For all $i = 1, \ldots, n$, the data point x_i projected onto this direction gives the principal component score z_{i1}.

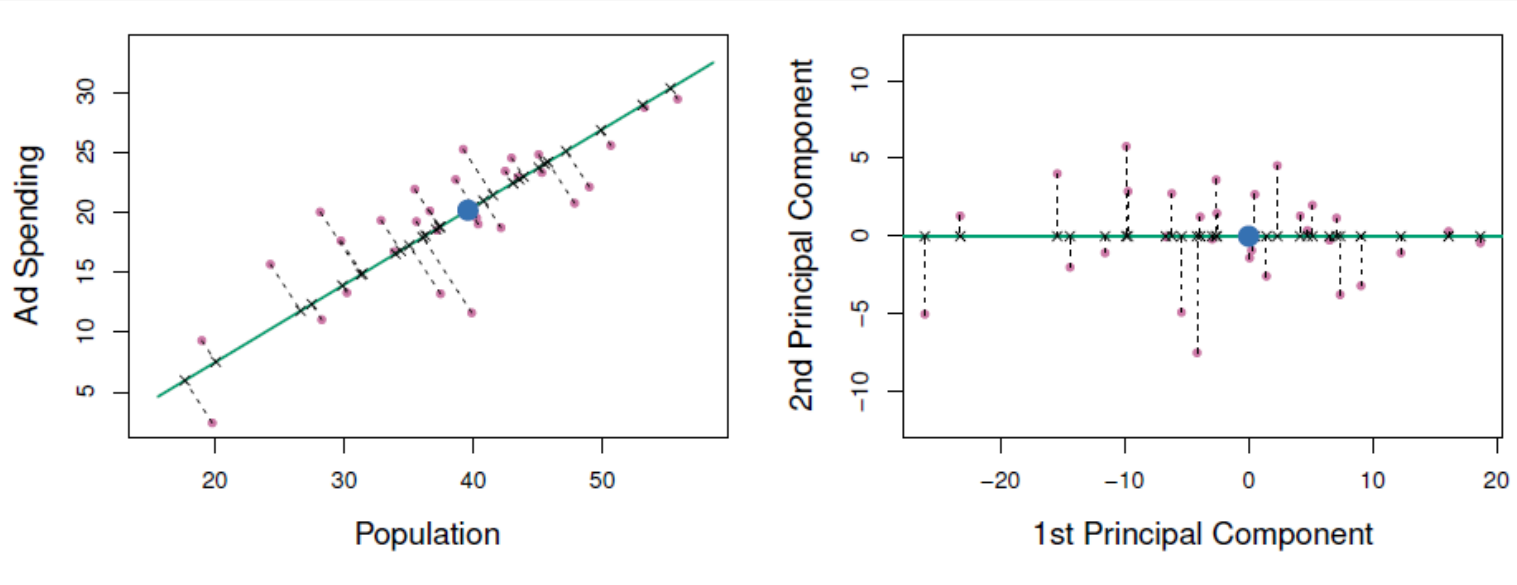

Figure 7: Figure by James et al. (2021). A subset of the advertising data. The mean pop and ad budgets are indicated with a blue circle. Left: The 1st principal component direction (in green). It is the dimension along which the data vary the most, and it defines the line that is closest to all n observations. Right: The left-hand panel has been rotated so that the 1st principal component direction coincides with the x-axis.

Remaining principal components

PCA computes the second principal component by again choosing the normalized linear combination of the p features that maximizes the sample variance, but now the linear combination must produce a loading vector that is orthogonal to the first principal component loading vector.

- If $p = 2$ and ϕ_1 is already determined, then there is only one direction orthogonal to ϕ_1, and we immediately obtain ϕ_2.
- If $p > 2$, there are infinitely many directions orthogonal to ϕ_1.
- PCA maximizes (4) with the additional constraint that ϕ_2 must be orthogonal to ϕ_1.
- Once ϕ_2 is computed, PCA finds ϕ_3 by maximizing (4) with the additional constraint that ϕ_3 has to be orthogonal to both ϕ_2 and ϕ_1.
- And so on to find ϕ_4, ϕ_5, etc.

Geometric interpretation: redux

Alternative interpretation: principal components provide low-dimensional linear surfaces that are closest to the observations.

- First PC loading vector is the line in p-dim space closest to the n obsns.
- First two PC loading vectors span the plane closest to the n obsns.
- First three PC loading vectors span the three-dim hyperplane closest to the n obsns. Etc.

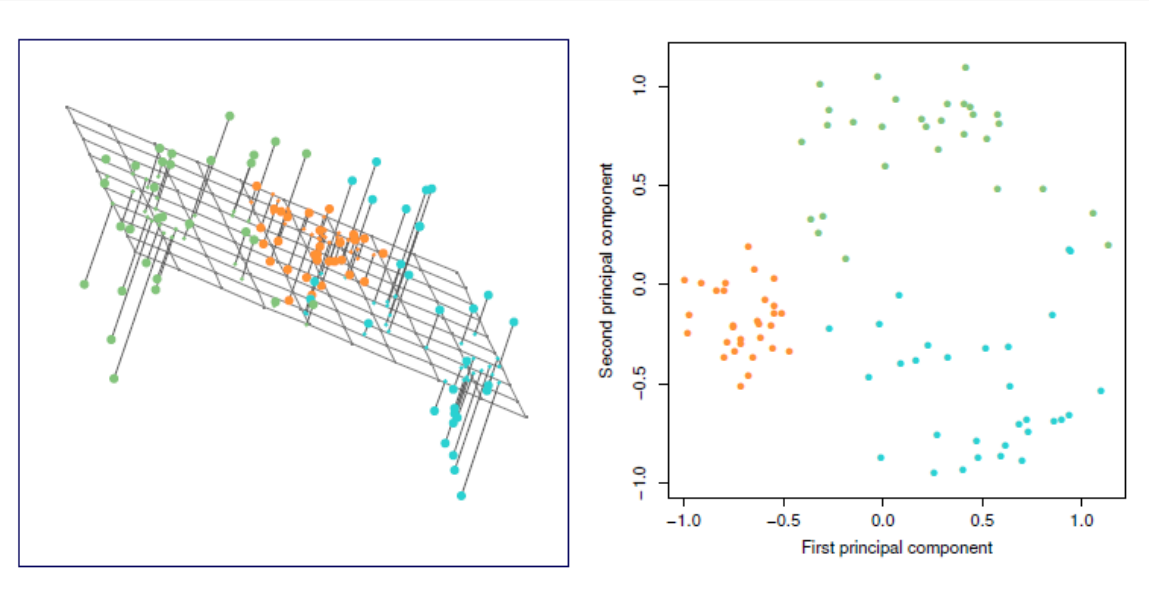

Figure 8: Figure by James et al. (2021). Observations simulated in three dimensions. The observations are displayed in color for ease of visualization. Left: The first two principal component directions span the plane that best fits the data (in sense of minimizing RSS). Right: The first two principal component score vectors give the coordinates of the projection of the observations onto the plane.

For any positive integer $M \leq \min\{n-1, p\}$, together the first M principal component score vectors and the first M principal component loading vectors provide an M-dimensional approximation to the ith observation:

$$x_{ij} \approx \sum_{m=1}^{M} z_{im}\phi_{jm}. \tag{5}$$

- How good is this approximation? It is the best, in the sense of the following optimization problem.
- Suppose we have a data matrix $\mathbf{X}$ that is column-centered, and have chosen some value of M.
- Of all possible approximations of the form $x_{ij} \approx \sum_{m=1}^{M} a_{im}b_{jm}$, consider which values $\{a_{im}\}$ and $\{b_{jm}\}$ minimize the sum of squares

$$\sum_{j=1}^{p}\sum_{i=1}^{n}\left(x_{ij} - \sum_{m=1}^{M} a_{im}b_{jm}\right)^2. \tag{6}$$

- It can be shown that for any value of M, the minimizers of (6) are exactly $a_{im} = z_{im}$ and $b_{jm} = \phi_{jm}$.
- When $M = \min\{n-1, p\}$, the representation (5) becomes exact.

The proportion of variance explained (PVE)

How much information is lost by projecting onto a few principal components, i.e., how much variance in the data is not contained in principal components?

- Assuming centered data, the *total variance* in the data is $\sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2$, and the variance explained by the mth principal component is $\frac{1}{n} \sum_{i=1}^{n} z_{im}^2$.

$$\underbrace{\sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2}_{\text{Var. of data}} = \underbrace{\sum_{m=1}^{M} \frac{1}{n} \sum_{i=1}^{n} z_{im}^2}_{\text{Var. of first } M \text{ PCs}} + \underbrace{\frac{1}{n} \sum_{j=1}^{p} \sum_{i=1}^{n} \left(x_{ij} - \sum_{m=1}^{M} z_{im} \phi_{jm} \right)^2}_{\text{MSE of } M\text{-dimensional approximation}}$$

- *Proportion of the variance explained* (PVE) of the mth principal component is the quotient of these values, so

$$PVE_m = \frac{\sum_{i=1}^{n} z_{im}^2}{\sum_{j=1}^{p} \sum_{i=1}^{n} x_{ij}^2} . \tag{7}$$

- Proportion of variance explained by the first M principal components is

$$\sum_{m=1}^{M} PVE_m = 1 - \frac{RSS}{TSS} , \tag{8}$$

where TSS is total sum of squared elements of $\mathbf{X}$, and RSS is residual sum of squares of the M-dimensional PC approximation.

Screeplot

PVE is usually visualized with the *screeplot*.

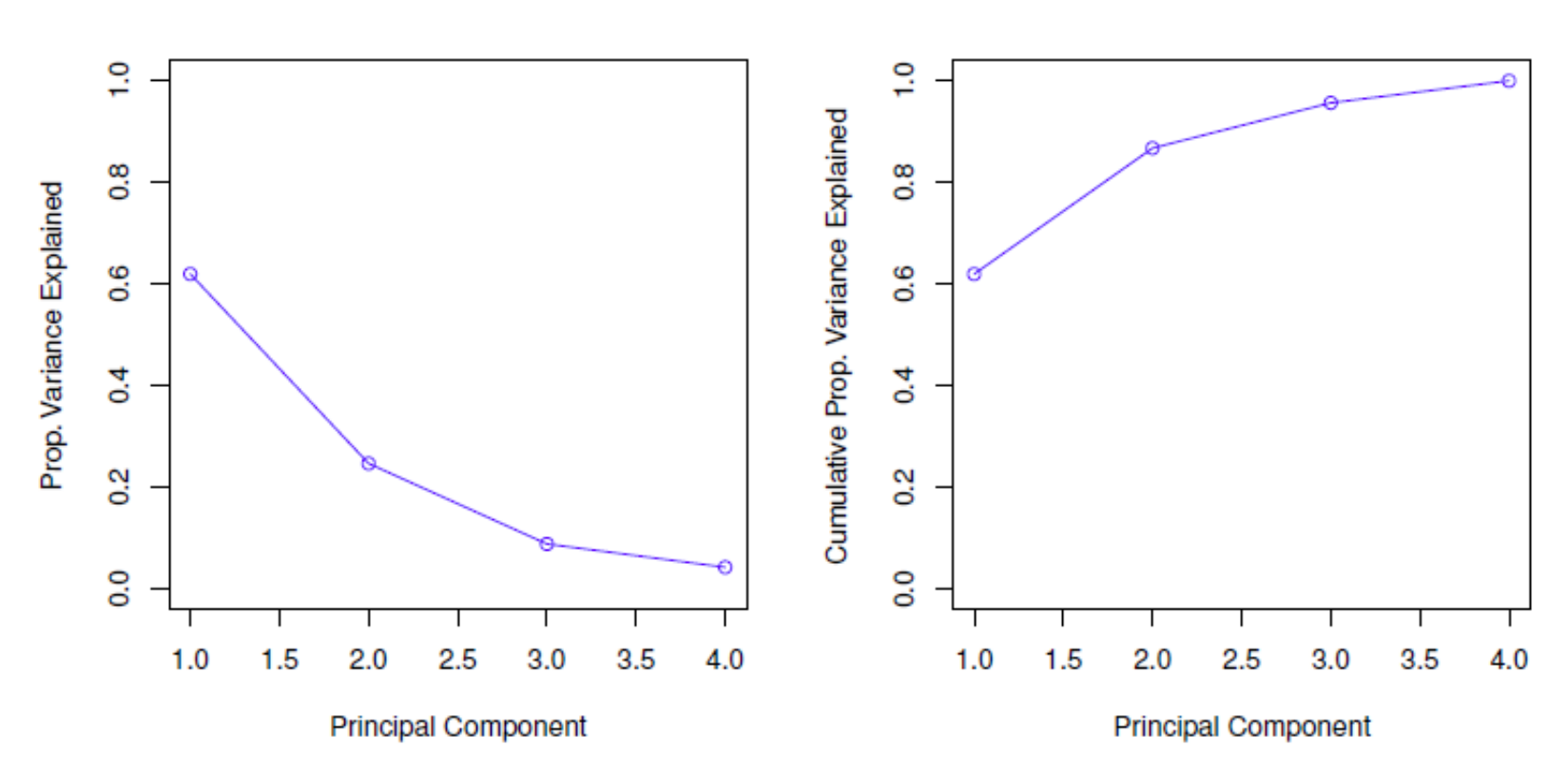

Figure 9: Figure by James et al. (2021). Left: A scree plot depicting the proportion of variance explained (PVE) by each of four principal components in USArrests data. Right: The cumulative proportion of variance explained by four principal components.

Number of predictors

- Usually, we only want to have a small dimension in order to better visualize or understand the data.
- How many principal components do we need? How can we justify to use only three instead of four or more principal components (e.g.)?
  - There is no simple answer to this! There is no formula that can be applied universally that gives us the optimum value.
  - We can intuitively decide to choose the number of predictors by eyeballing the *screeplot* which depicts the proportion of variance explained (PVE).
  - In the screeplot, we look for a point at which the PVE by each subsequent principal component drops significantly off (this is subjective). Such a drop is often referred to as an *elbow* in the screeplot, and the rule by choosing the point is thus called *elbow rule*.

Scaling the variables

We are usually interested to analyze the impact of certain features in relation to their variation.

- The variance of features can be large solely because their values are.
- Not an issue if features are measured in the same units.
- So, by scaling (by their standard deviation), the variation among all predictors is comparable, independently of their magnitude.
- In general, scaling the variables to have standard deviation one is recommended.

Other uses for PCA

Many statistical techniques, such as regression, classification, and clustering, use the full $n \times p$ data matrix.

- Can instead use the $n \times M$ matrix whose columns are the first $M \ll p$ principal component score vectors.
- This can lead to less noisy results, since often the signal in a data set is concentrated in its first few principal components.