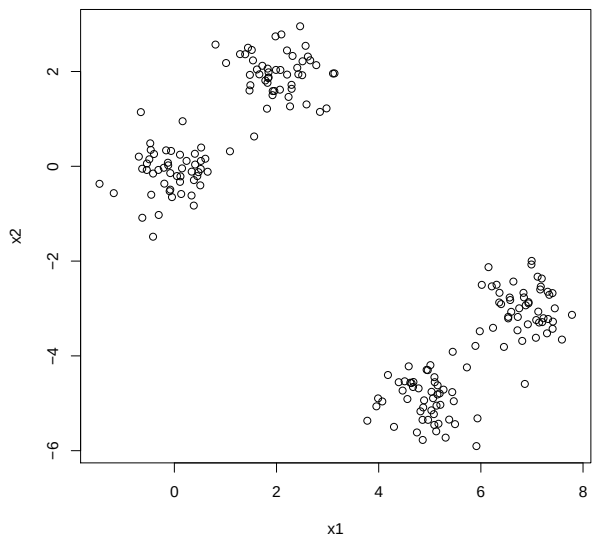
MSA220 - Statistical Learning for Big Data

Lecture 13

Rebecka Jörnsten

Mathematical Sciences
University of Gothenburg and Chalmers University of Technology

Explorative analysis - finding groups in data.
This is a more difficult task than classification since the goal is rather subjective - what is group?



Are there 2 or 4 clusters?

What defines a group is up to you to choose, e.g. by defining an object-object similarity measure or distance.
The most commonly used distance measure is euclidean distance.
However, other distances may be more appropriate to use for some data sets, e.g. matching-metrics for categorical data or correlation-based similarities for curve data or when relative differences between features are more interesting than absolute levels.

Main traditional approaches

1. Partitioning methods: kmeans, PAM
2. bottom-up methods: hierarchical clustering
3. Model-based methods/density based methods

How to generate a partition

kmeans is a very popular method and has been around for a long time. It is very simple and fast.

1. Pick K observations at random to be the cluster representatives or *centroids*, $\mu_k, k = 1, \cdots, K$.
2. Allocate observations i to the cluster whose centroid it is closest to
$$C(i) = \arg\min_k d(x_i, \mu_k),$$
where $d(x_i, \mu_k)$ is the distance between observation location x_i and centroid μ_k.
3. Update the centroids as
$$\mu_k = \sum_{C(i)=k} x_i / N_k, \quad N_k = \sum_{i=1}^{n} 1\{C(i) = k\}$$
4. Iterate until convergence (usually very fast).

Note, you may have to run the algorithm a couple of times to ensure you have converged to a local optimum due to poor choice of initial centroids.

Hierarchical clustering

Hierarchical clustering is very popular since it is simple, intuitive and comes with a nice graphical display. Like PAM, it takes pairwise distances as input which makes it rather flexible.
In contrast to PAM and kmeans it constructs clusters "bottom-up", i.e. building clusters by joining observations together as opposed to splitting the data into groups ("top-down").

Hierarchical clustering

1. Start with all the observations as their own clusters, $g_1, g_2, \cdots, g_n$, each cluster of size 1.
2. Join the pair of clusters g_i and g_j that are the closest together
3. Keep on joining clusters pairs until all observations are in one big clusters of size n.

Step 2 involves some subjective choices:
what is close? that is, what kind of distance metric do you want to use?
what is meant by clusters being close? that is, how do we combine information about observation pairwise distances into a group-level distance?

Linkage

Cluster-cluster distance is called *linkage*

- **average linkage** is the most commonly used. The distance between clusters g and h is computed as

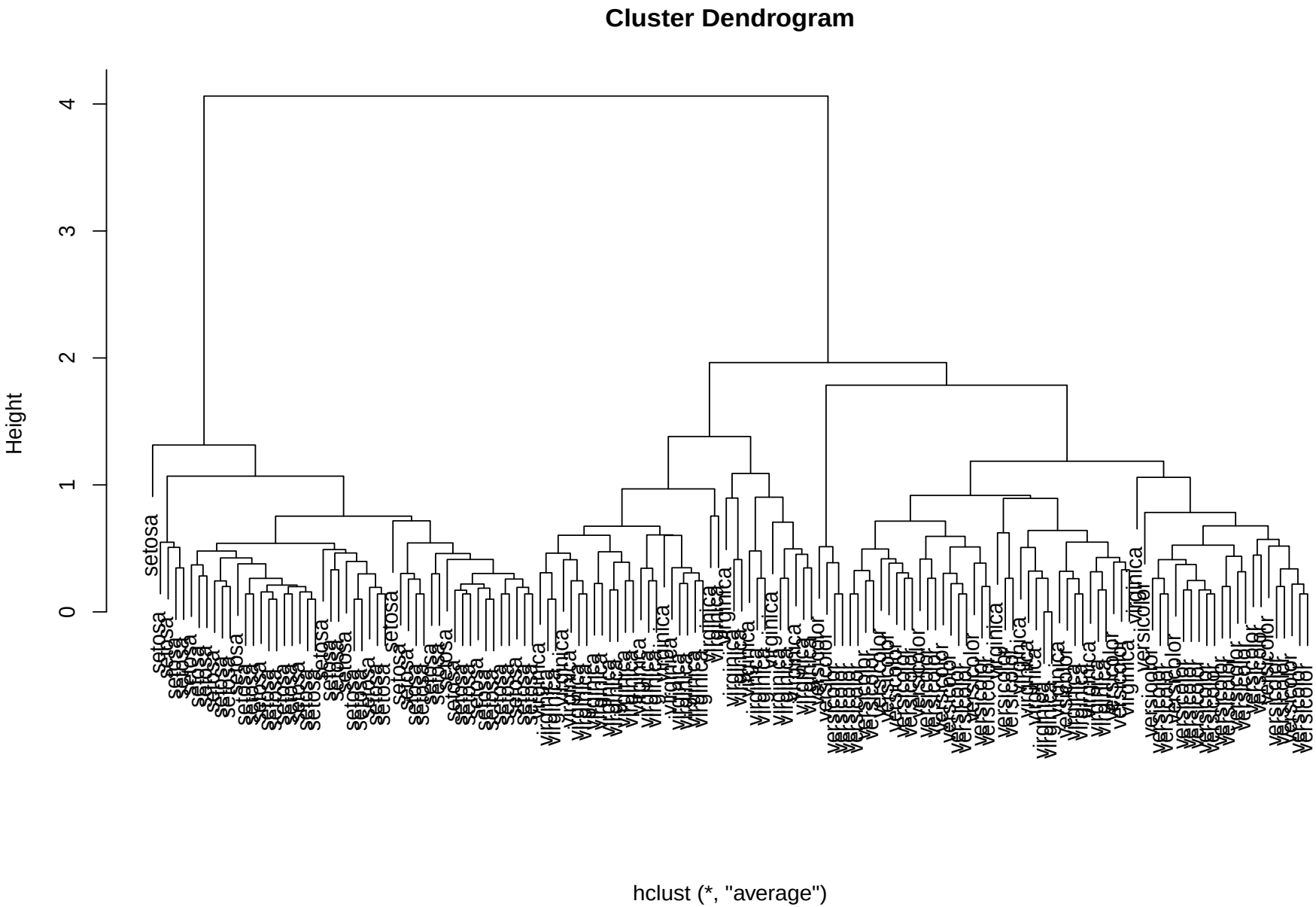
$$d_{gh} = \sum_{i:C(i)=g, j:C(j)=h} d_{ij} / \sum_{i:C(i)=g, j:C(j)=h} 1$$

  The average similarity between all pairs in the two different clusters is encouraged.

Dendrogram

Hierarchical clustering is graphically summarized with the *dendrogram*. This depicts the iterative procedure of joining clusters.
The dendrogram looks a bit like a CART tree but the meaning is different. You read the dendrogram from the bottom-up, this is how the clusters are formed. The length of the branches in the dendrogram represents the cluster-cluster distances. A really long branch indicates that the within-cluster distances were increased a lot by joining the cluster at the bottom of the branch with the other cluster at the top of the branch.
The dendrogram can therefore suggest how many clusters you should form from your data. Long branches can be cut to form distinct group that have small within-cluster distance and is well separated from the rest of the observations.

Hierarchical clustering



The average linkage identifies two groups of irises (long branches in the dendrogram): setosa and the versicolor/virginica. The latter group is very mixed up.

Modelbased clustering

So far we have looked at nonparametric cluster methods - clusters are defined through a distance metric and a construction algorithm/criterion. We have noted that clustering is a difficult problem because these choices are subjective.
Parametric, or modelbased clustering, takes clustering into a familiar statistical modeling framework where we can say something about the goodness-of-fit of clusters. It is a statistical problem that can be objectively analyzed BUT of course relies on a modeling assumption that is a subjective choice nonetheless.

The multivariate normal assumption sounds a bit like discriminant analysis. The difference here is that we don't know the class label! We have already looked at a similar problem when we talked about the mixture discriminant analysis method where the class labels were known but the component-labels within each class was not. There we solved the problem with the EM-algorithm, and that is what we do here as well and I will give you a bit more detailed info as well.

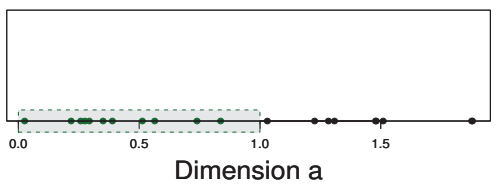
EM-iterations

- If we knew the labels, we could estimate the parameters of each cluster easily just like we did in discriminant analysis.
- If we knew the model parameters, we could allocate observations to each cluster using the posterior probability approach, just like in discriminant analysis.

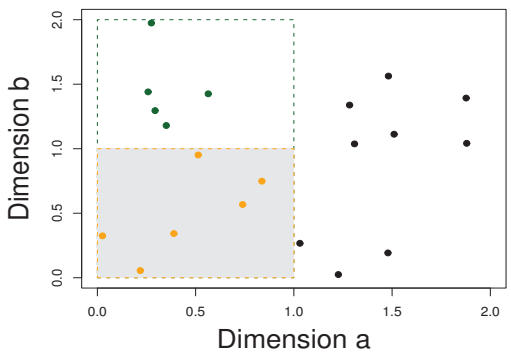
This iterative process is the EM approach to model fitting and is a method used to solve complex estimation problems that would be easy to solve with some additional information (as done in each step or the iterative procedure).

High-dimensional clustering

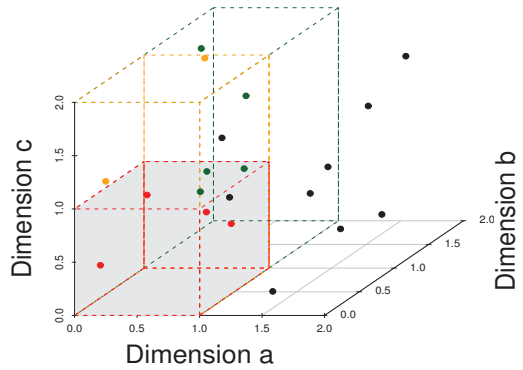
What goes wrong when the data is high-dimensional?



(a) 11 Objects in One Unit Bin



(b) 6 Objects in One Unit Bin

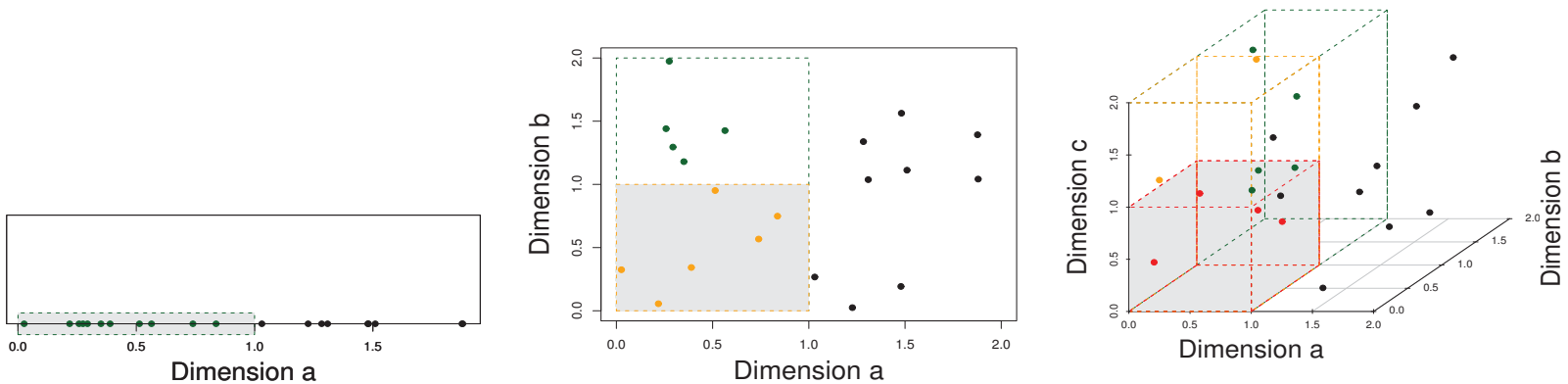


(c) 4 Objects in One Unit Bin

Figure 1: The *curse of dimensionality*. Data in only one dimension is relatively tightly packed. Adding a dimension stretches the points across that dimension, pushing them further apart. Additional dimensions spreads the data even further making high dimensional data extremely sparse.

High-dimensional clustering

The notion of similar and dissimilar brakes down - everyone is far apart in high-dimensional space!
Clustering is all about distances - and the concept of relative distance brakes down.



(a) 11 Objects in One Unit Bin (b) 6 Objects in One Unit Bin (c) 4 Objects in One Unit Bin

Figure 1: The *curse of dimensionality*. Data in only one dimension is relatively tightly packed. Adding a dimension stretches the points across that dimension, pushing them further apart. Additional dimensions spreads the data even further making high dimensional data extremely sparse.

High-dimensional clustering

What to do?

- Feature selection
- Feature transformation

Feature selection

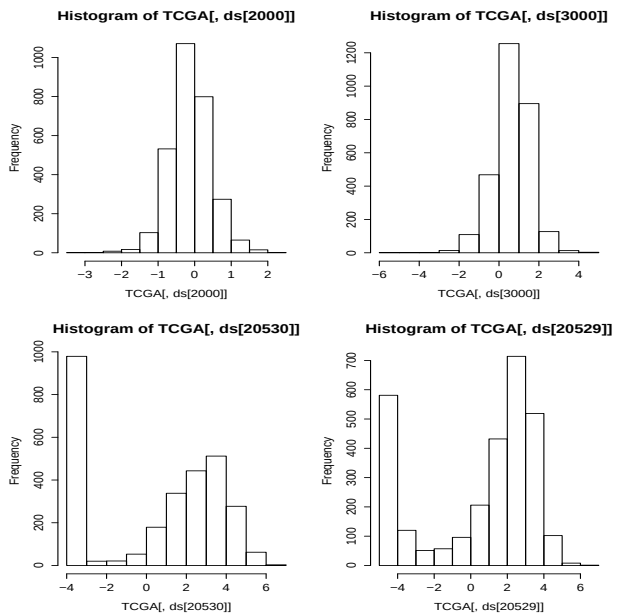
- This is a much easier task for high-dimensional classification
- For example, run ANOVA on each feature and choose the most significant features to train the classifier on
- How can we screen for clustering strength when we don't know the clusters?

Feature selection

- Take the most variable features.
- The idea is that large spread is due to cluster separation
- Caution: this is a bad idea if features are measured at different scales!

Feature selection

- An alternative is to think that a cluster feature should have a clear multi-modal distribution where each "hump" in the distribution corresponds to a cluster
- Screen features by testing for unimodality (Hartigan's dip-test).
- Keep features with the largest test statistic against unimodality



Feature transformation

- We can also transform the data - projecting onto a lower-dimensional space
- We want to ensure we retain as much information as possible
- PCA to the rescue!
- Keep principal components corresponding to the largest eigenvalues

Dimension reduction and clustering

- Careful! Check how sensitive the results are to your screening
- both the type of screening and
- how aggressively you screen

Consensus clustering

- Any method that comprises many steps is subject to instability since each step is a source of error
- How many features, how many eigenvalues?
- In addition, many clustering methods are quite sensitive to small data perturbations

Consensus clustering

- If you can do things once, you can do it 100 times!
- Add some randomness to the procedure and run it many times
- Retain clusters that are *stable* across multiple runs!

Consensus clustering

- How add randomness?
- Subset of features
- Subset of features + PCA
- Random projections
- ....

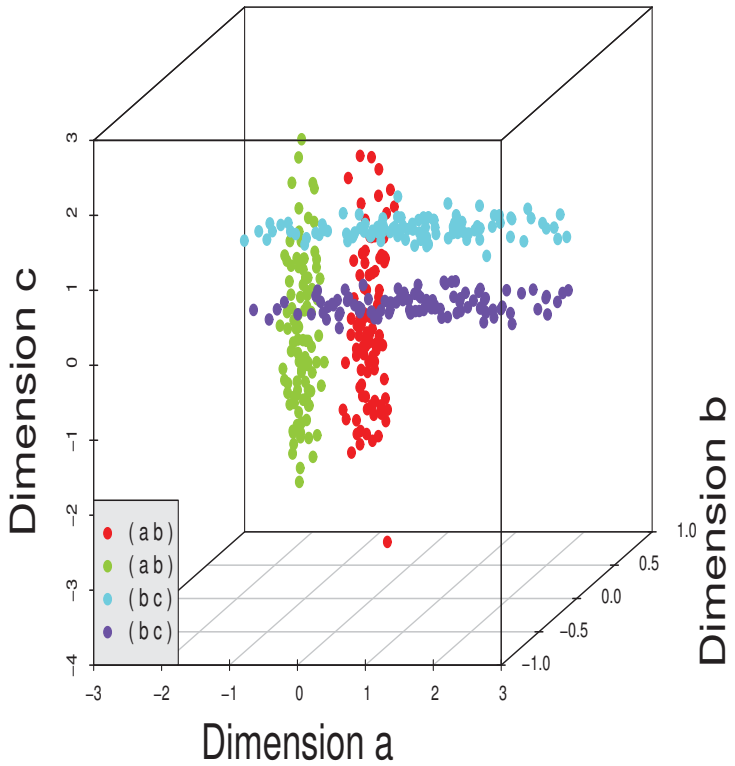
Consensus clustering

- Each run produces a clustering result
- How do we combine these?
- Some methods compare the clusters in terms of overlap
- Other methods use a similar idea to RF clustering: for each pair of objects, count how many times they appear in a cluster together. Use this is a new similarity metric and use e.g. hierarchical clustering to produce a final result.
- I like the latter approach because it gives you a lot of flexibility in which clustering procedures to compare across runs.

Subspace clustering

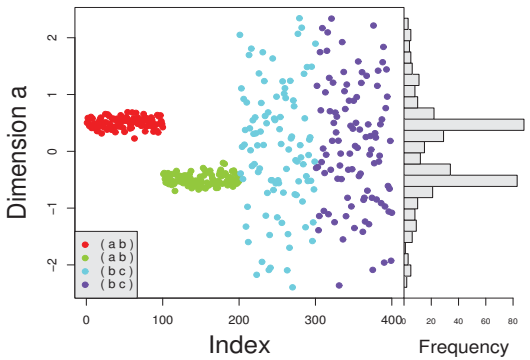
- Another method for dealing with high-dimensional data
- Assume each cluster only "lives" in a subspace (subset) of dimensions
- If we knew which subspace we could adjust how we compute distances and circumvent the COD (curse of dimensionality)

Subspace clustering

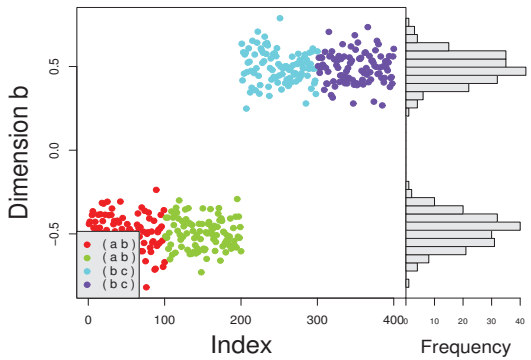


- Here are some nice figures from sigkdd review paper (see class home page) of Parsons, Hague and Liu
- 4 clusters that live in different subspaces

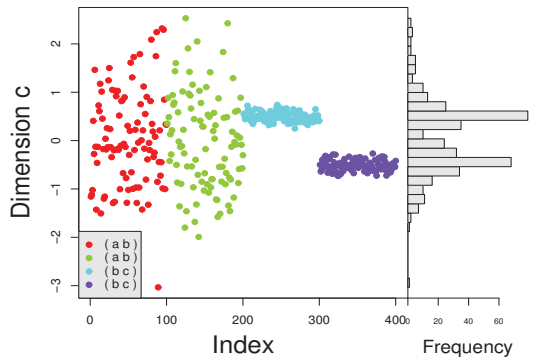
Subspace clustering



(a) Dimension a

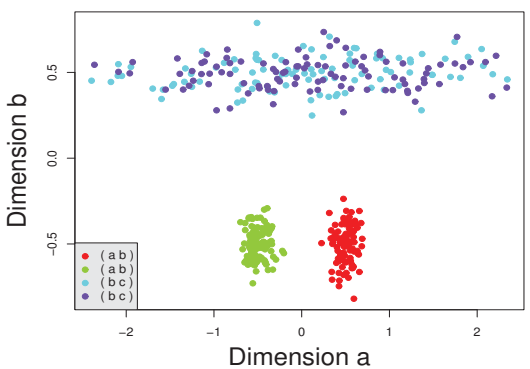


(b) Dimension b

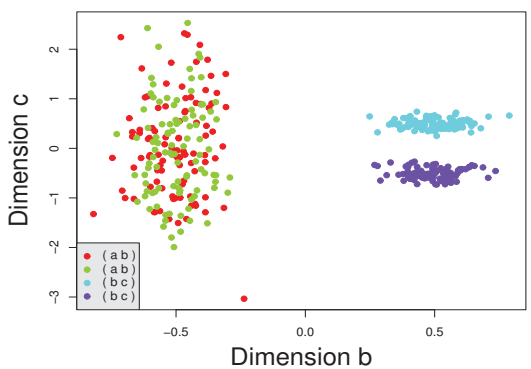


(c) Dimension c

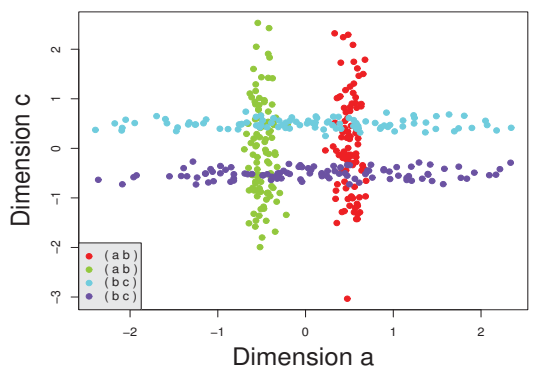
Figure 3: Sample data plotted in one dimension, with histogram. While some clustering can be seen, points from multiple clusters are grouped together in each of the three dimensions.



(a) Dims a & b

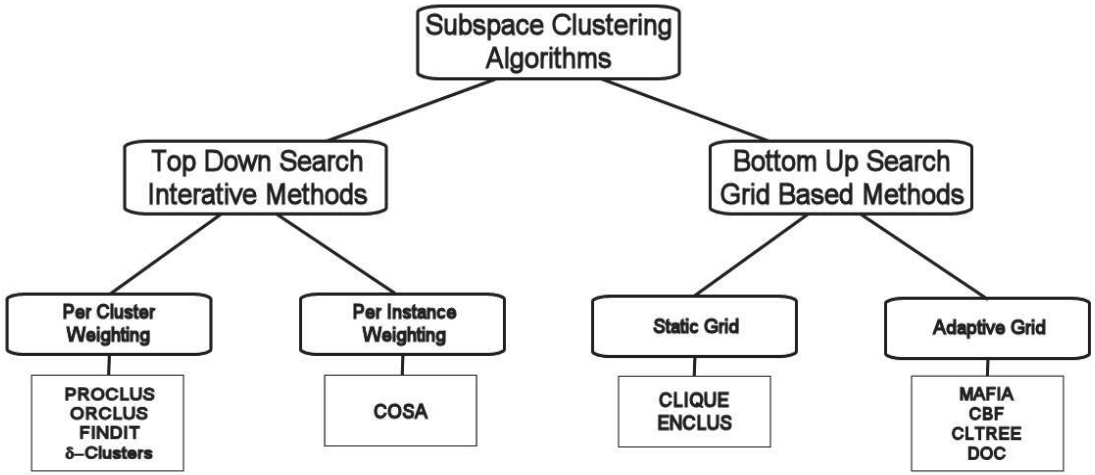


(b) Dims b & c



(c) Dims a & c

Subspace clustering



- In subspace clustering there are also two main approaches
- Bottom-up/Grid-based
- Top-down search

Subspace clustering

An example of a bottom-up method is CLIQUE

- Generate a grid in high-dimensional space by dividing each dimension into say 10 equal length intervals
- Each high-dimensional rectangle now contains a set of observations
- We search for a connected set of dense rectangles in a subspace

Subspace clustering

CLIQUE

- Empty or near empty rectangles are removed, the density threshold tau is a tuning parameter
- For each set of two dimensions we check if there are two neighboring dense units in these two dimensions and then they are saved as a cluster.
- This is repeated for all sets of three, four, five,. . . dimensions. After every step adjacent clusters are replaced by a joint cluster.

Subspace clustering

Top-down methods work along these lines (there are many methods so check out the posted paper). Here I outline the PROCLUS method

- Start with a larger than desired set of randomly selected "medoids" that are far apart in the data
- We're now going to iteratively update a k-medoid clustering
- We select k medoids at random.
- We check for each medoid if it any good: i.e. is it at a center of densely clustered observations:
- We check in which subdimension the cluster lives by looking at within-cluster distances as a function of subspace dimension. We assign observations to the subspace medoid and if very few observations are allocated to it we remove this medoid and choose another observation at random as a new seed.

Subspace clustering

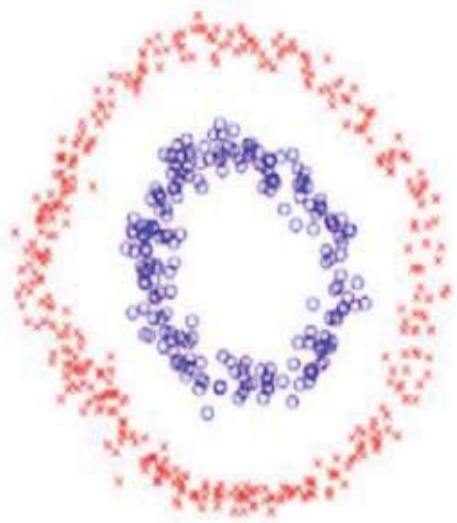
Subspace clustering outputs clusters and their subspace dimension

- A way to deal with complex structures and high-dimensions
- Can also be interesting to interpret clusters in terms of their subspaces
- Ongoing research and applications to video, images, genomics,...

Spectral clustering

- Compactness, e.g., k-means, mixture models
- Connectivity, e.g., spectral clustering

Compactness



Connectivity

https://charlesmartin14.wordpress.com/2012/10/09/spectral-clustering/

- Most clustering methods are geared at finding dense, compact regions
- What if clusters are more complex than that?

Spectral clustering

Similarity graphs

- A similarity measure between observational pairs can be illustrated with a graph
- The length of an edge between objects inversely proportional to the similarity
- If we threshold similarities that are small we get a graph where only some observations are connected
- Graph-partitioning: which edges should we cut to form good clusters? Clearly the ones with low similarity.

Spectral clustering

Similarity graphs

- $w_i j$ is the adjacency graph edge between object i and j
- $d_i = \sum_j w_{ij}$ is the *degree* of node i
- If we partition the graph into node sets A and B the "cost" of this operation is $\sum_{i \in A, j \in B} w_{ij}$
- A good graph-partitioning minimizes this cost

Spectral clustering

Spectral clustering is a fast a simple method that produces a graph-cut

- Form the adjacency matrix W and the degree matrix D
- Define the Laplacian $L = D - W$
- Fact: For any vector f: $f'Lf = \sum_{i,j}^{n} w_{ij}(f_i - f_j)^2$
- Fact: The eigenvalues of L: $0 = \lambda_1 \leq \lambda_2 \leq \cdots \lambda_n$

Spectral clustering

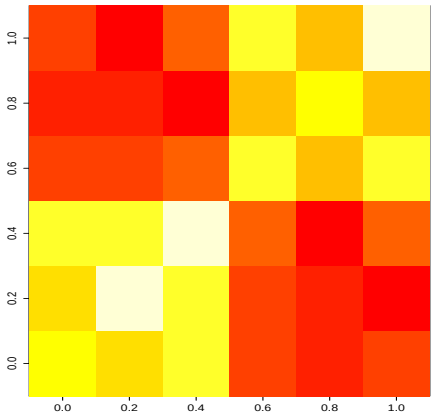
- Turns out: if there are k connected components in your graph (clusters) then there are k zero-eigenvalues of L
- and the corresponding eigenvectors can be used to find the clusters using e.g. kmeans.

Spectral clustering

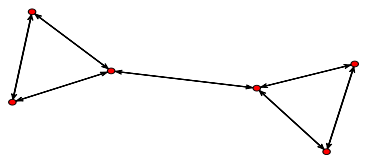
- Special case: one connected component
- Assume f is an eigenvector with value 0
- $0 = f'Lf = \sum w_{ij}(f_i - f_j)^2$ means we have have $f_i = f_j$ for all $w_ij > 0$
- That is, f has to be constant in any part of the graph that is connected!
- Now think about a case with k connected components
- The corresponding eigenvector has to be constant for all objects that are connected!

Spectral clustering

Simple example: block-diagonal similarity matrix



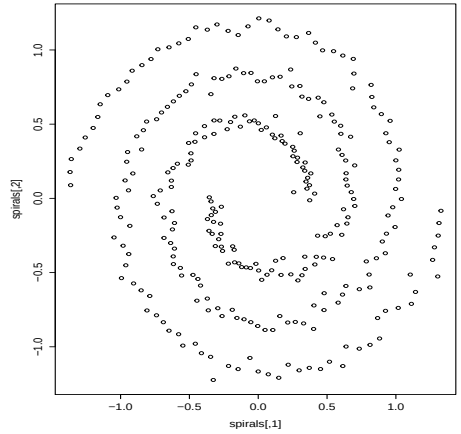
corresponding graph

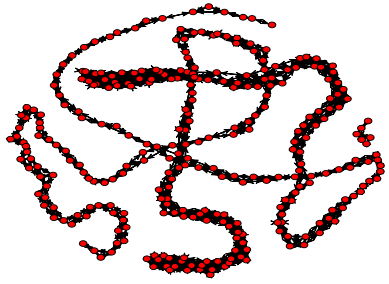


and leading eigenvectors $(0, 0, 0, .57, .57, 57)$ and $(.57, .57, .57, 0, 0, 0)$

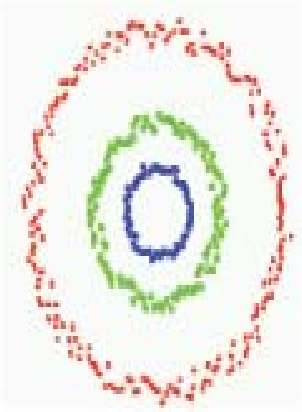
Spectral clustering

More complex example:

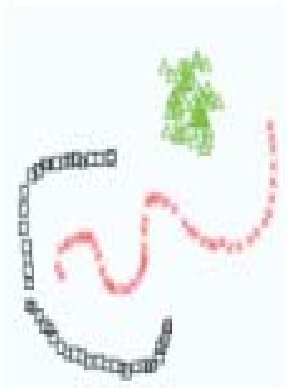




Spectral clustering



OR



$$L = \begin{pmatrix} L_1 & & 0 \\ & L_2 & \\ 0 & & L_3 \end{pmatrix}$$

$$\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$$

First three eigenvectors

Spectral clustering

- Compute a similarity matrix S
- From this, compute the graph or adjacency matrix W (e.g. by thresholding the similarities)
- Compute the Laplacian L
- Compute the eigenvalues of L - look for a good separation between small and large values
- Cluster the eigenvectors corresponding to the smallest eigenvalues using kmeans.

Project 3

Doodle!

1. Selecting the number of clusters.
   - We have looked at Silhouette width, cluster prediction strength and BIC (for model based clustering) but there are many more
   - Search for other methods ("elbow"-type methods are kind of like the F-test, there are those based on simulating a "null" (no clusters), those that use entropy.
   - Try out methods on 2-3 data sets and discuss.
2. Selecting the number of clusters.
   - Use for example the elbow-method for kmeans or silhouette width
   - Investigate impact of; i) sample size, ii) dimensionality; iii) noise level; iv) number of irrelevant features,...

Project 3

3. Method comparisons
   - Using a select set of methods from class and 2-3 methods we haven't discussed
   - Apply to 3 data sets and discuss differences: interpretability, stability, ease-of-use,...
4. Small clusters and big
   - Can clustering methods find small clusters among big ones?
   - Check the literature for methods to handle unbalanced data
   - Try on simulated data - which methods work and which do not?

Project 3

5 Dimension reduction and clustering
- Using SVD and NMF
- Compare on 2-3 data sets how well clustering works if you use feature transformation

6 Dimension reduction and clustering
- Random projections
- Compare on 2-3 data sets. Pros and Cons.

Project 3

7 Clustering and variable selection
- For model based clustering, a few different methods exist (clustvarsel for example)
- But you can also use screening and filtering and subspace clustering
- on 2-3 data sets (simulated or real), try variable selection and discuss results

8 Consensus clustering
- Check the literature for a few variants on consensus clustering
- Apply to 2-3 data sets
- Discuss and interpret

Project 3

9 Clustering and big sample size
- Some methods scale better with respect to sample size than others (and some with respect to dimension)
- Check literature for big-n clustering methods
- Apply to 2-3 data sets and discuss

10 Non-linear methods
- SVD,PCA comprise key components of many methods, but they are linear representations of data
- *non-linear PCA* is a more flexible summary. Check the literature for this and related methods.
- Apply to 2-3 data sets and discuss pros and cons.