Clustering

Yee Whye Teh
Gatsby Computational Neuroscience Unit
UCL

Adaptive Modelling of Complex Data
UCL

Outline

Adaptive Modelling of Complex Data

We cover:

	Supervised learning	Unsupervised learning
Discrete output/latents	**Classification**	**Clustering**
Continuous output/latents	**Regression**	**Dimensionality reduction**

and: **Time-Series models**

We will not cover:

- Other learning paradigms:
  - Reinforcement learning
  - Semi-supervised learning
  - ...
- Other data domains:
  - Relations
  - Strings
  - Graphs
  - ...

Outline

What is Clustering?

It is what you think it is.

- We naturally put things into categories, or clusters.
- People, movies, organisms...

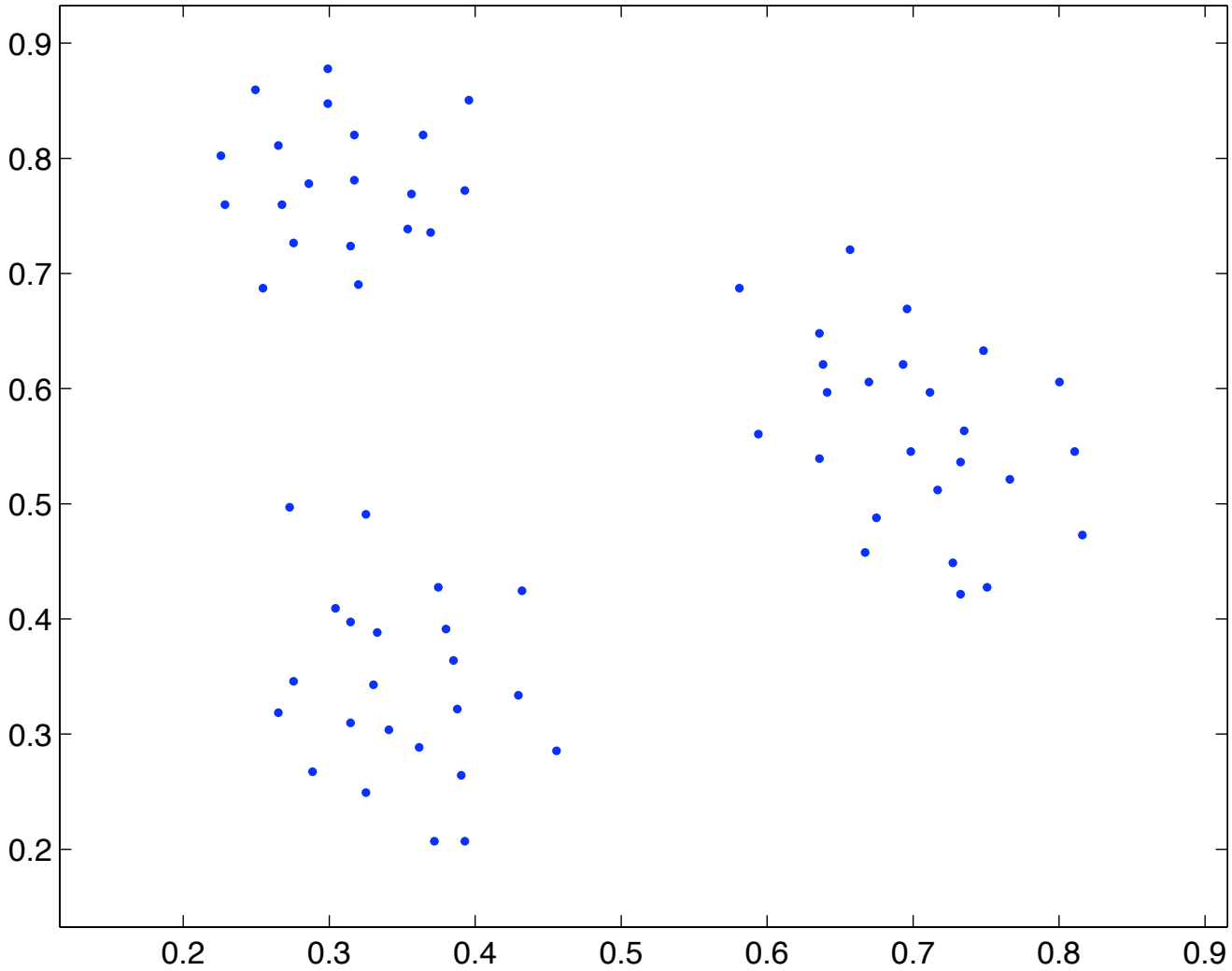
An Impossibility Theorem for Clustering

Jon Kleinberg
Department of Computer Science
Cornell University
Ithaca NY 14853

Abstract

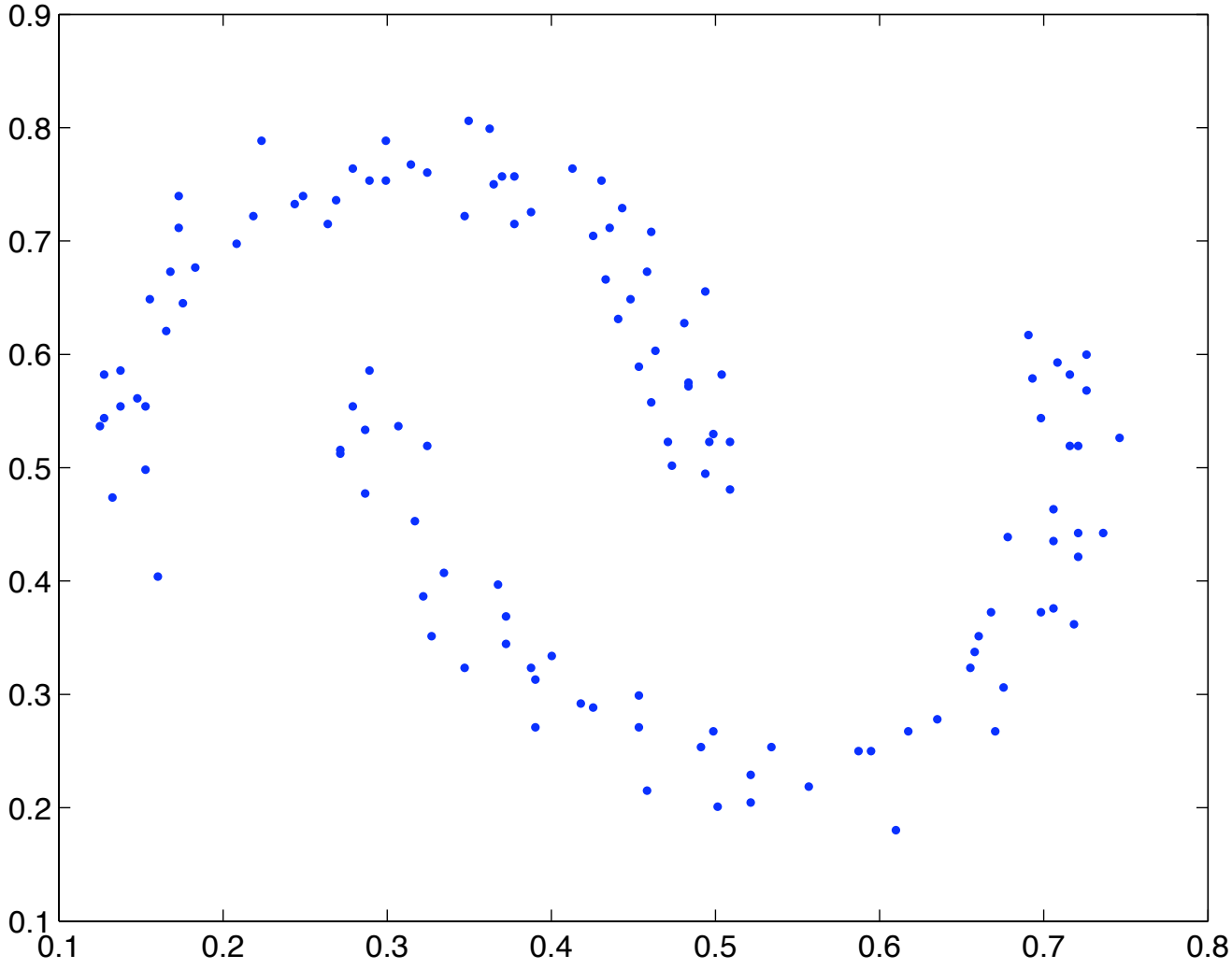
Although the study of *clustering* is centered around an intuitively compelling goal, it has been very difficult to develop a unified framework for reasoning about it at a technical level, and profoundly diverse approaches to clustering abound in the research community. Here we suggest a formal perspective on the difficulty in finding such a unification, in the form of an *impossibility theorem*: for a set of three simple properties, we show that there is no clustering function satisfying all three. Relaxations of these properties expose some of the interesting (and unavoidable) trade-offs at work in well-studied clustering techniques such as single-linkage, sum-of-pairs, k-means, and k-median.

What is Clustering?



- Partitioning or grouping data into "**similar**" subsets.

What is Clustering?



- Partitioning or grouping data into "**similar**" subsets.

Formalizing Clustering

- Given data vectors $\mathbf{x}_1, \ldots, \mathbf{x}_m$, a K-clustering is an assignment $c_i \in \{1, \ldots, K\}$ of each data vector $\mathbf{x}_i$ to a cluster c_i.
- We can also represent this using a 1-of-K coding:

$$r_{ic} = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ is assigned to cluster } c, \\ 0 & \text{otherwise.} \end{cases}$$

  Note $r_{ic} \geq 0$ and $\sum_c r_{ic} = 1$.
- Each cluster c may be described using a set of parameters θ_c.
- We use an objective function to measure quality of clustering:

$$J(\boldsymbol{\theta}, R)$$

- Clustering is the process of optimizing the objective function:

$$\operatorname*{argmin}_{\boldsymbol{\theta}, R} J(\boldsymbol{\theta}, R)$$

Notation

m	Number of data vectors.
n	Number of dimensions of data vectors.
K	Number of clusters.
$\mathbf{x}_1, \ldots, \mathbf{x}_m$	Data vectors.
c_i	Cluster index of data vector $\mathbf{x}_i$.
r_{ic}	Does $\mathbf{x}_i$ belong to cluster c?
μ_c	Prototype vectors.
Ψ_c	Variability of data vectors around prototype.

Outline

K-means

- K-means is a **prototype** based clustering algorithm.
- The prototype for the c'th cluster is μ_c.
- Each data vectors will belong to exactly one cluster, say:

$$r_{ic} = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ belongs to cluster } c, \\ 0 & \text{otherwise.} \end{cases}$$

- *How do we find good prototypes, and good assignments of data vectors to prototypes?*

K-means

Assigning Data Vectors to Clusters

- **Suppose: we have good prototypes.**
- *How do we assign data vectors to clusters?*
- Easy: assign data vectors to closest prototype!
- For data vectors $i = 1, \ldots, m$, for prototypes $c = 1, \ldots, K$:

$$d_{ic} = \|\mathbf{x}_i - \mu_c\|^2$$

$$c_i = \operatorname*{argmin}_c d_{ic}$$

$$r_{ic} = \begin{cases} 1 & \text{if } c_i = c, \\ 0 & \text{otherwise.} \end{cases}$$

K-means

Finding Good Prototypes

- **Suppose: we have good assignments of data vectors to cluster.**
- *How do we find good prototypes?*
- Easy: let the prototypes be the means of each cluster!

$$\mu_c = \frac{\sum_{i=1}^{m} r_{ic} \mathbf{x}_i}{\sum_{i=1}^{m} r_{ic}}$$

K-means

Finding Good Prototypes and Assignments

- We are faced with a *chicken-and-egg* problem, since we do not have good prototypes nor assignments to begin with.
- Solution: iterate until prototypes and assignments stabilize.
- Objective function:

$$J(R, \boldsymbol{\mu}) = \sum_{i=1}^{m} \sum_{c=1}^{K} r_{ic} \|\mathbf{x}_i - \mu_c\|^2$$

- Iterations:

$$R \Leftarrow \operatorname*{argmin}_{R} J(R, \boldsymbol{\mu})$$

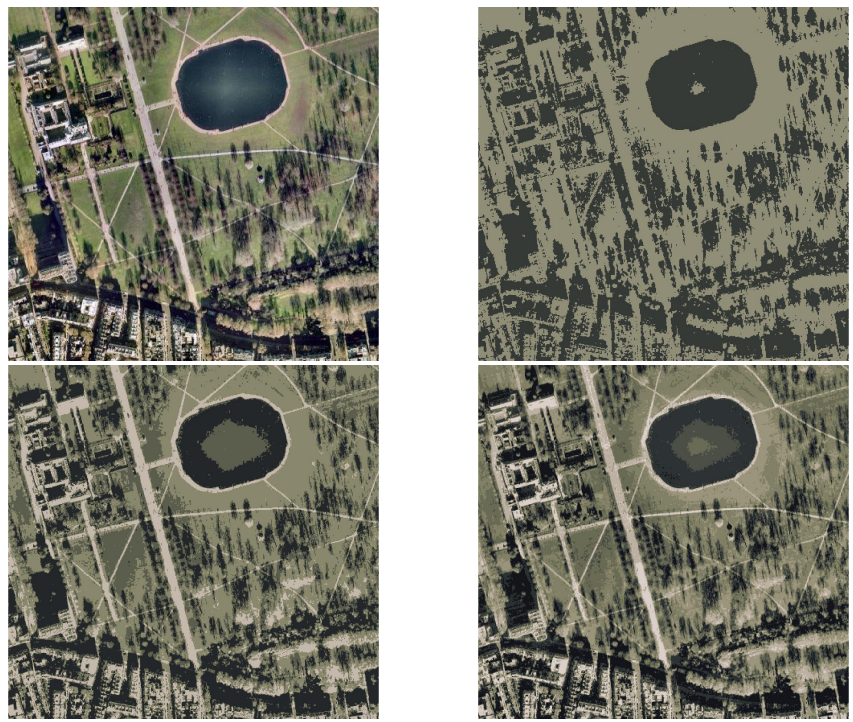
$$\boldsymbol{\mu} \Leftarrow \operatorname*{argmin}_{\boldsymbol{\mu}} J(R, \boldsymbol{\mu})$$

- Demonstration.

K-means

Applications

- *Summarization*: replace data vector with cluster label.
- *Lossy compression*: store prototypes and cluster labels (**vector quantization**).
- *Image segmentation*: e.g. cluster pixel colours in images.



Small patches, visually relevant features, and *spectral clustering* improve results.

K-means

Extensions

- Other distance measures.
  E.g. cityblock, cosine, correlation, Hamming, kernels.
- K-mediods.
  Use a data vector as the cluster prototype: makes sense when means are either expensive or not well-defined.
- Mixture models, spectral clustering, hierarchical clustering.
  Rest of course.

K-means

Issues

- Local minima.
  Different initializations of K-means can lead to different solutions.
  Solution: Run multiple times and use best run.
- Empty clusters.
  Solution: either drop empty clusters, or re-use them elsewhere.
- Finding an appropriate K.
  Objective function is of no help: increasing K always decreases J.

$$J(R, \boldsymbol{\mu}) = \sum_{i=1}^{m} \sum_{c=1}^{K} r_{ic} \|\mathbf{x}_i - \mu_c\|^2$$

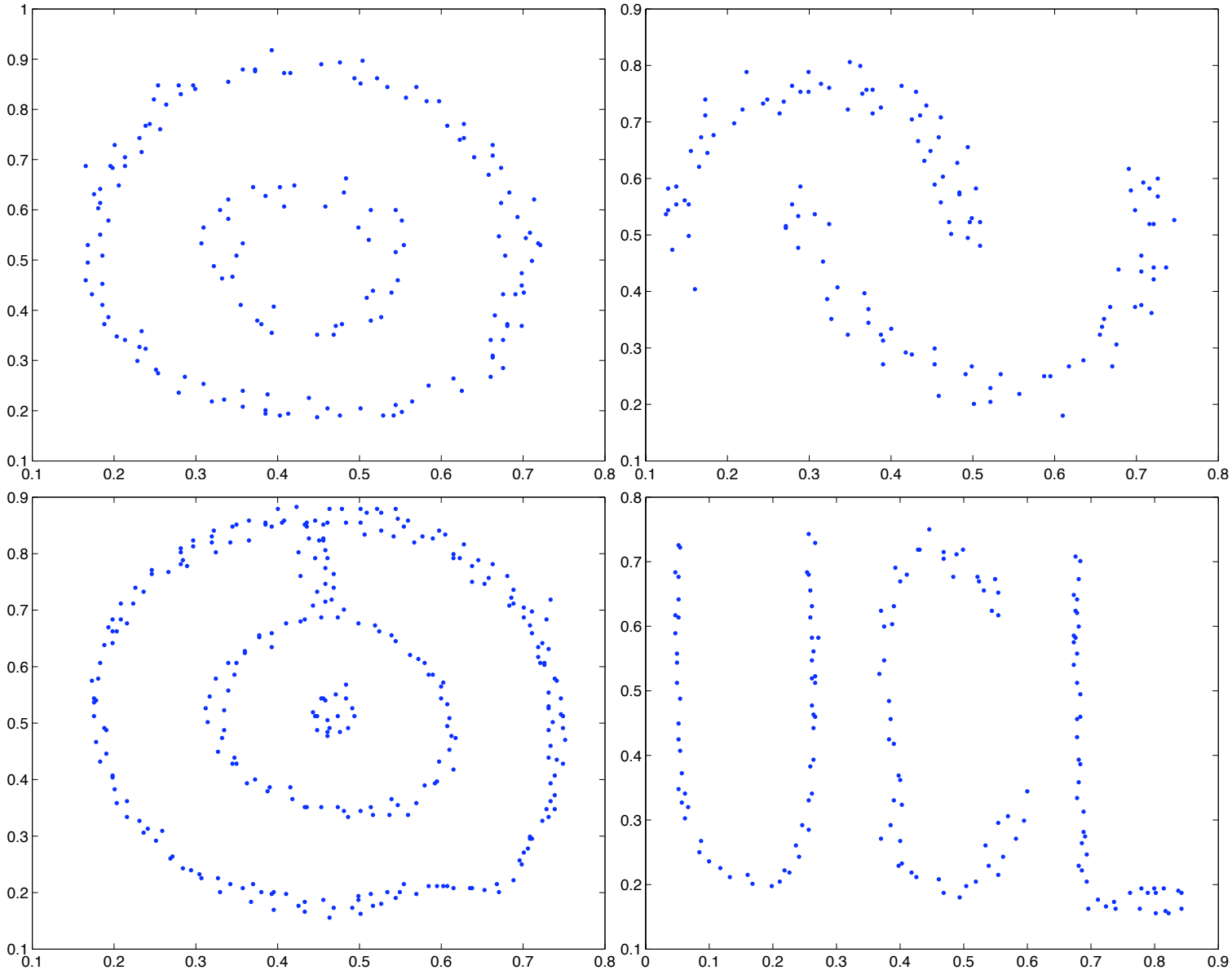
Solutions?

- Correlate clusterings with external data, e.g. additional labels.
- Minimum description length.
- Bayesian probabilistic approaches.

Outline

Spectral Clustering

Stranger Clusters



Spectral Clustering

Stranger Clusters

- For these clusters, the shape of the clusters is unimportant.
  This rules out K-means or any prototype based model.
- What is important is similarity of data vectors to other data vectors in the same cluster.
  Similarities are propagated transitively.

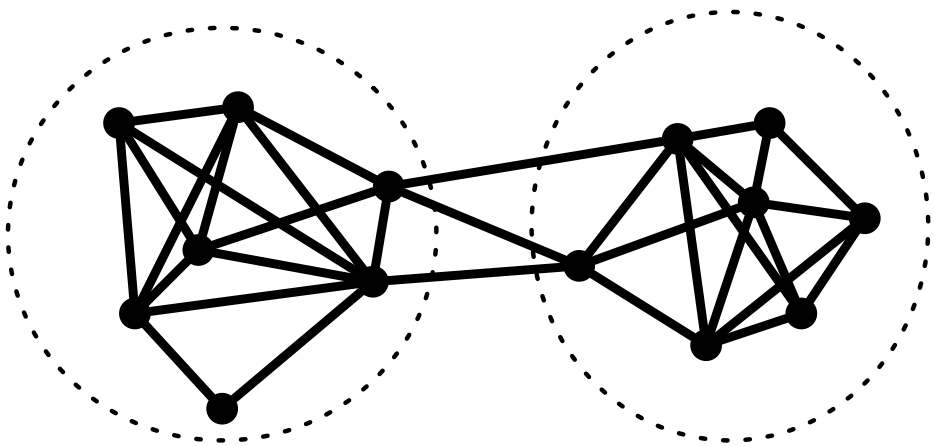
$$\mathbf{x}_1 \sim \mathbf{x}_2 \text{ and } \mathbf{x}_2 \sim \mathbf{x}_3 \Rightarrow \mathbf{x}_1 \sim \mathbf{x}_3$$

- This implies using a matrix of *similarities* between data vectors, and algorithms operating on such similarity matrices.

Spectral Clustering

Graph Partitioning Approaches

- We can formalize similarities between data points using graphs.
- Data items are vertices of the graph, and edges connect similar data items.
- If similarities are graded, we can attach a weight W_{ij} (similarity score) to each edge ij instead.
- Clusters are **highly connected** components of the graph.



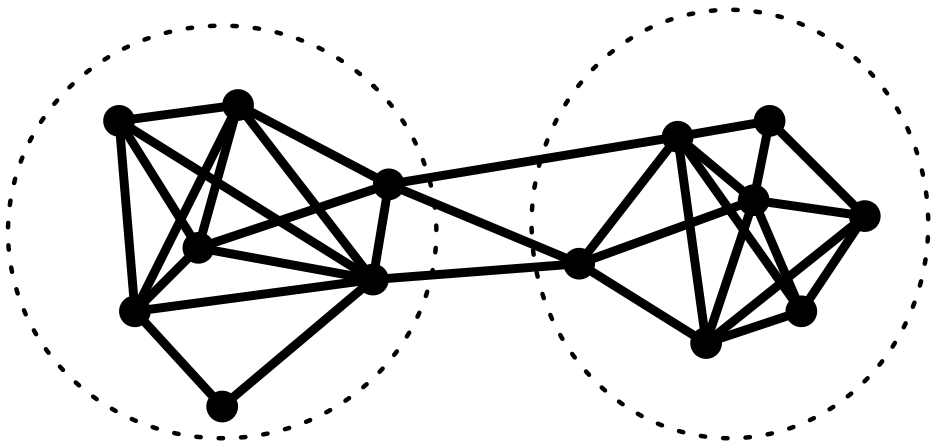
Spectral Clustering

Normalized Cuts

- Let C and D be a partition of the vertices into two clusters.
- An obvious approach is to find C and D minimizing the **cut**:

$$\text{cut}(C, D) = \sum_{i \in C} \sum_{j \in D} W_{ij}$$

  the total weight of edges between C and D (cut by the partition).
- This does not work well because it often finds single vertices for one of the clusters.



[Shi and Malik 2000]

Spectral Clustering

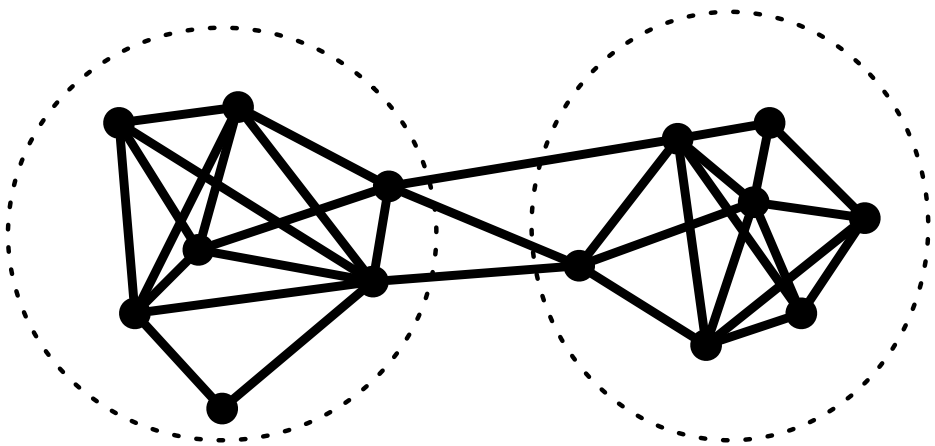
Normalized Cuts

- We can prevent singleton clusters by normalizing for the sizes of the clusters in some way.
- The **normalized cut** is defined to be:

$$\text{ncut}(C, D) = \frac{\text{cut}(C, D)}{\text{assoc}(C, V)} + \frac{\text{cut}(C, D)}{\text{assoc}(D, V)}$$

where $$\text{assoc}(A, B) = \sum_{i \in A} \sum_{j \in B} W_{ij}$$

each fraction is the ratio of total weight to the other cluster versus total weight of all edges originating from the cluster.



Spectral Clustering

Normalized Cuts

- Finding the partition minimizing the normalized cut can be expressed as a discrete optimization problem:

$$\underset{\mathbf{y}}{\operatorname{argmin}} \frac{\mathbf{y}^\top (D - W)\mathbf{y}}{\mathbf{y}^\top D\mathbf{y}}$$

  where $\mathbf{y}$ is a vector with each entry corresponding to a vertex, constrained to take on only the values $\{1, -b\}$ for some $b > 0$, and D is a diagnonal matrix

$$D_{ii} = \sum_{j \in V} W_{ij}$$

- If we forget that entries of $\mathbf{y}$ can only take on values $\{1, -b\}$, and allow $\mathbf{y}$ to be an arbitrary vector, the above is *exactly* the same objective function for **Laplacian Eigenmaps**.

Spectral Clustering

Normalized Cuts

- Laplacian Eigenmap:

$$\underset{\mathbf{y}}{\operatorname{argmin}} \frac{\mathbf{y}^\top (D - W)\mathbf{y}}{\mathbf{y}^\top D\mathbf{y}}$$

- A d dimensional embedding is obtained from the smallest $d + 1$ generalized eigenvectors of the system

$$(D - W)\mathbf{y} = \lambda D\mathbf{y}$$

- The intuition is that these generalized eigenvectors are the modes of vibrations of the system described by the graph (e.g. edges are springs with varying stiffness and vertices are balls).
- If the data were clustered, the graph has densely connected components, and the modes of vibration will be *precisely* the clusters.

Spectral Clustering

Normalized Cuts

- Compute similarities and construct the W matrix of similarities.
- Compute the diagonal matrix D.
- Find the 2nd smallest generalized eigenvector of the system:

$$(D - W)\mathbf{y} = \lambda D\mathbf{y}$$

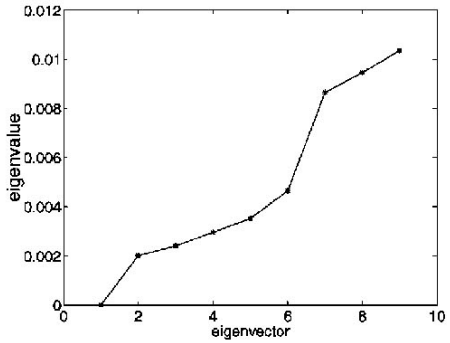
- Find a value c so that vertices i with $y_i > c$ form a good cluster, and likewise those with $y_i < c$.

Spectral Clustering

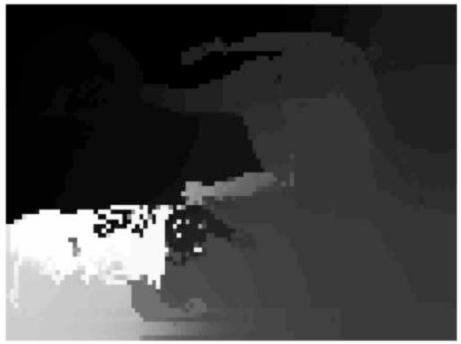
Normalized Cuts

Spectral Clustering

Normalized Cuts



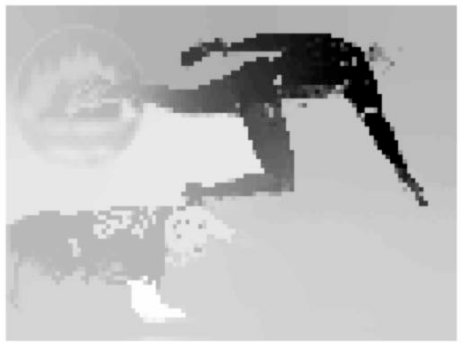
(a)



(b)

(c)

(d)

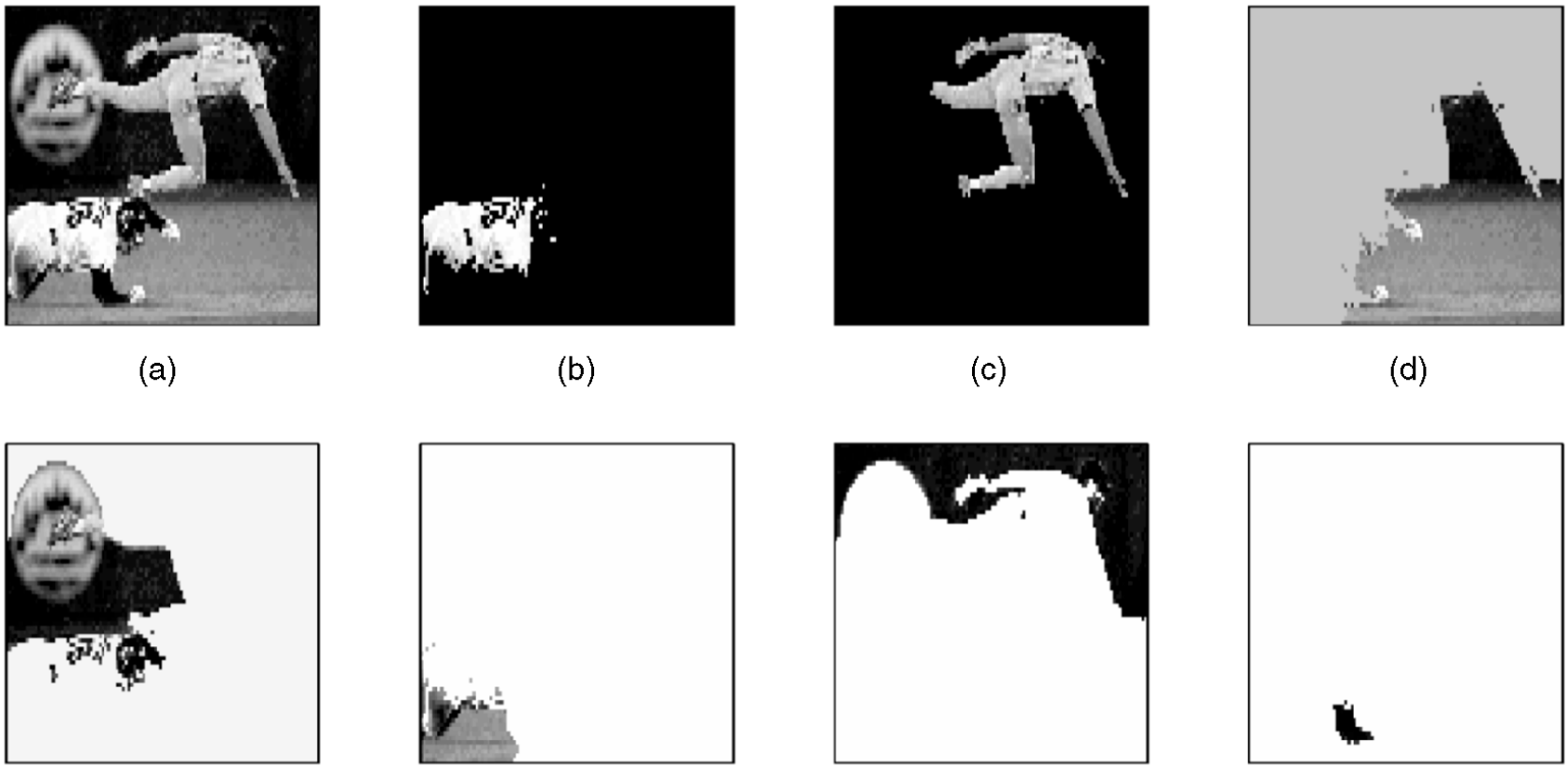


(e)

(f)

Spectral Clustering

Normalized Cuts



Spectral Clustering

A Second Algorithm

- Compute the W and D matrices.
- Find the eigenvectors corresponding to the K largest eigenvalues of $D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$. Form matrix Y whose columns are the eigenvectors.
- Normalize the rows: $\tilde{Y}_{ij} = Y_{ij}/(\sum_j Y_{ij}^2)^{\frac{1}{2}}$.
- Perform K-means on the rows of $\tilde{Y}$.
- Assign $\mathbf{x}_i$ to cluster c if the i'th row of $\tilde{Y}$ is assigned to cluster c.

The eigensystem here is just a negated and rotated form of the previous eigensystem (which is why we find the K largest eigenvectors instead of smallest).
Vibration story: if clusters are well separated, then the smallest eigenvalues are all 0, one for each cluster. Eigenvectors will be rotationally invariant so clustering using all K eigenvectors better.

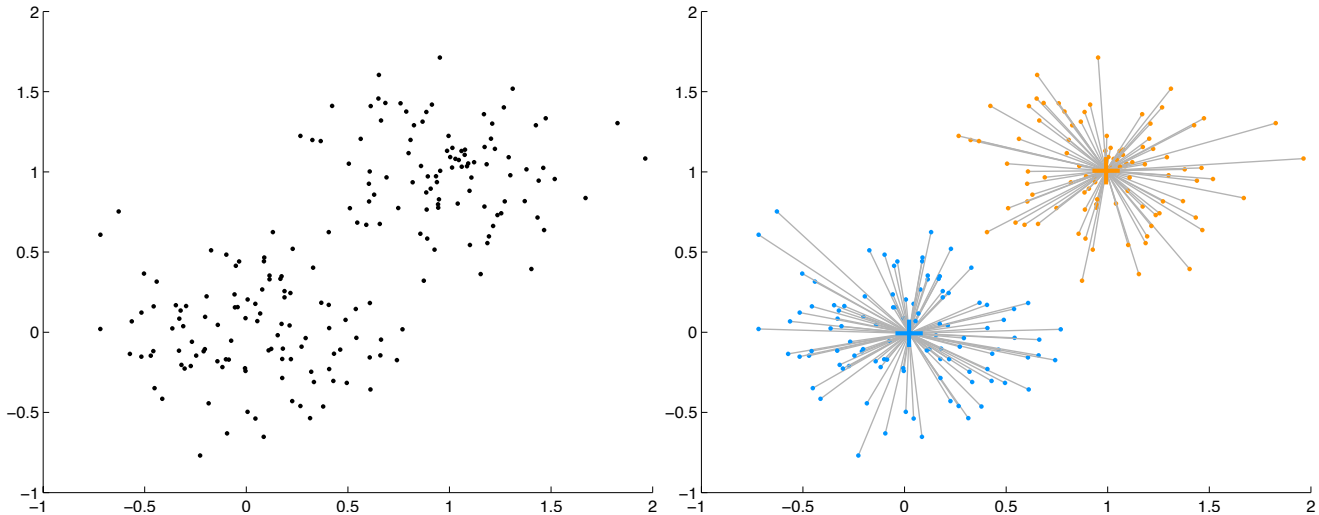
[Ng, Jordan and Weiss 2001]

Outline

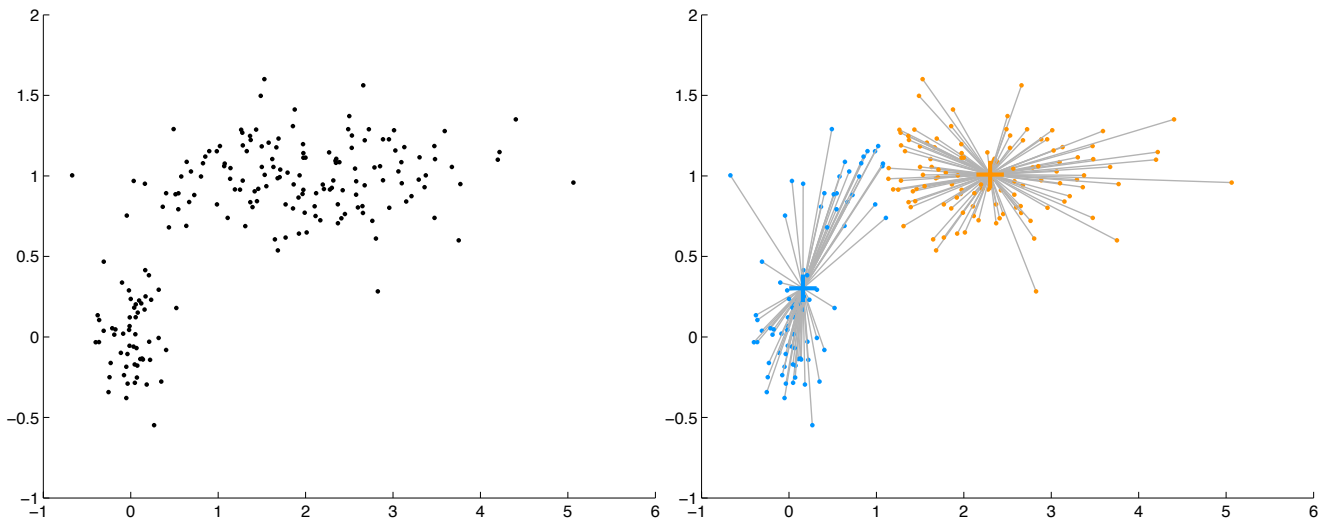
Mixture Models

Issues with K-means

- Overconfidence in assignment of data vectors to clusters.



- Did not take into account variability of clusters.

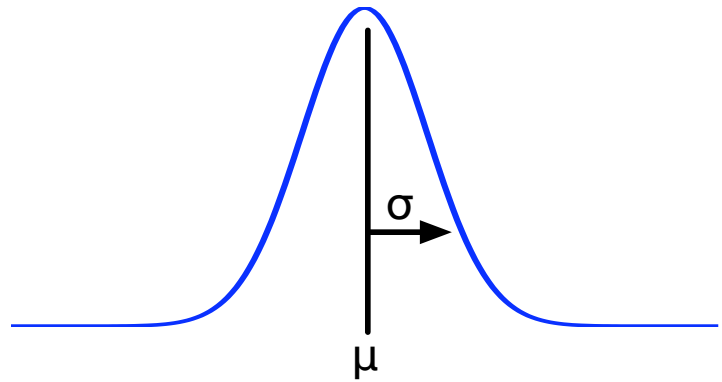


Mixture Models

Modelling Variability with Gaussians

- Gaussians are the most commonly encountered distributions in probability and statistics.

$$\mathcal{N}(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



- Multidimensional Gaussians

$$\mathcal{N}(\mathbf{x};\mu,\Psi) = |2\pi\Psi|^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^\top \Psi^{-1}(\mathbf{x}-\mu)}$$

μ	Mean (centre) of the Gaussian.
Ψ	Width (variability) of the Gaussian in different directions.

Mixture Models

Modelling Variability with Gaussians

- Instead of representing each cluster by only its prototype (mean), we also represent variability in the cluster using multidimensional Gaussians.
- What are good means and covariances of Gaussians, given assignments of data vectors to clusters?

$$\mu_c = \frac{\sum_{i=1}^{m} r_{ic} \mathbf{x}_i}{\sum_{i=1}^{m} r_{ic}}$$

$$\Psi_c = \frac{\sum_{i=1}^{m} r_{ic} (\mathbf{x}_i - \mu_c)(\mathbf{x}_i - \mu_c)^\top}{\sum_{i=1}^{m} r_{ic}}$$

- These optimal parameters are known as **maximum likelihood** parameters.

Mixture Models

Modelling Uncertainty in Cluster Assignments

- The probability of a data vector under a Gaussian:

$$\mathcal{N}(\mathbf{x}_i; \mu_c, \Psi_c) = \frac{1}{\sqrt{|2\pi\Psi_c|}} e^{-\frac{1}{2}(\mathbf{x}_i - \mu_c)^\top \Psi_c^{-1}(\mathbf{x}_i - \mu_c)}$$

  Gives a measure of how likely is it that the data vector belongs to the cluster.

- We can use this to give confidence weighted estimates of cluster assignments:

$$r_{ic} = \frac{\mathcal{N}(\mathbf{x}_i; \mu_c, \Psi_c)}{\sum_{k=1}^{K} \mathcal{N}(\mathbf{x}_i; \mu_k, \Psi_k)}$$

$r_{ic} =$ Probability that data vector i belongs to cluster c.

Mixture Models

Iterative Algorithm

- Compute **responsibility** of clusters over data vectors:

$$r_{ic} = \frac{\rho_c \mathcal{N}(\mathbf{x}_i; \mu_c, \Psi_c)}{\sum_{k=1}^{K} \rho_k \mathcal{N}(\mathbf{x}_i; \mu_k, \Psi_k)}$$

- Update parameters of Gaussians given responsibilities:

$$\mu_c = \frac{\sum_{i=1}^{m} r_{ic} \mathbf{x}_i}{\sum_{i=1}^{m} r_{ic}}$$

$$\Psi_c = \frac{\sum_{i=1}^{m} r_{ic} (\mathbf{x}_i - \mu_c)(\mathbf{x}_i - \mu_c)^\top}{\sum_{i=1}^{m} r_{ic}}$$

- Update relative cluster sizes (mixing proportions):

$$\rho_c = \frac{\sum_{i=1}^{m} r_{ic}}{m}$$

Mixture Models

Probabilistic View of Mixture Models

- A mixture model is a **probabilistic** model.
  It defines a distribution over data vectors and cluster assignments.
- Probability of assigning $\mathbf{x}_i$ to cluster c:

$$p(y_i = c|\boldsymbol{\rho}) = \rho_c$$

- Probability of $\mathbf{x}_i$ given it is in cluster c:

$$p(\mathbf{x}_i|y_i = c, \mu_c, \Psi_c) = \mathcal{N}(\mathbf{x}_i; \mu_c, \Psi_c)$$

- Joint probability over everything:

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_m, y_1, \ldots, y_m|\boldsymbol{\rho}, \boldsymbol{\mu}, \boldsymbol{\Psi}) = \prod_{i=1}^{m} \prod_{c=1}^{K} (\rho_c \mathcal{N}(\mathbf{x}_i; \mu_c, \Psi_c))^{\mathbb{I}(y_i=c)}$$

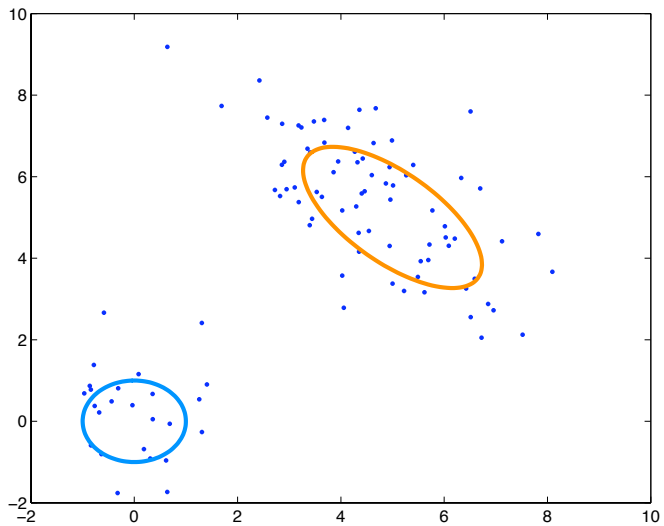
Mixture Models

Mixture Models as Generative Models

- Such a probabilistic model is **generative**—it describes a process of generating data sets. Example:

$$\begin{bmatrix}\rho_1\\ \rho_2\end{bmatrix} = \begin{bmatrix}.3\\ .7\end{bmatrix} \qquad \mu_1 = \begin{bmatrix}0\\ 0\end{bmatrix} \qquad \Psi_1 = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

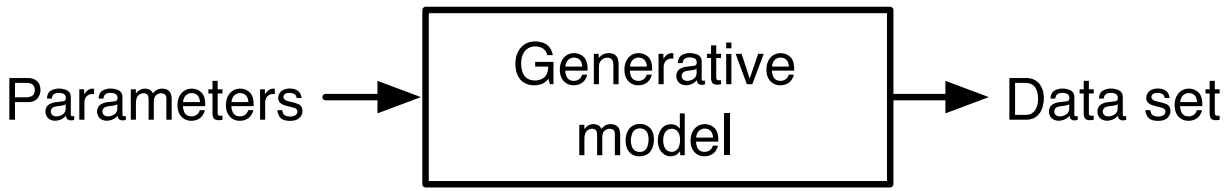
$$\mu_2 = \begin{bmatrix}5\\ 5\end{bmatrix} \qquad \Psi_2 = \begin{bmatrix}3 & -2\\ -2 & 3\end{bmatrix}$$



Mixture Models

Maximum Likelihood and the EM algorithm

- A *generative model* describes a process of *generating* data from a parametrized probabilistic model.



- **Maximum likelihood** is an approach to *recovering* parameters given data.

- Straightforward (in principle): find parameters such that the probability of generating the given data set is maximized.
- The **Expectation-Maximization** algorithm finds parameters that locally maximizes the likelihood.

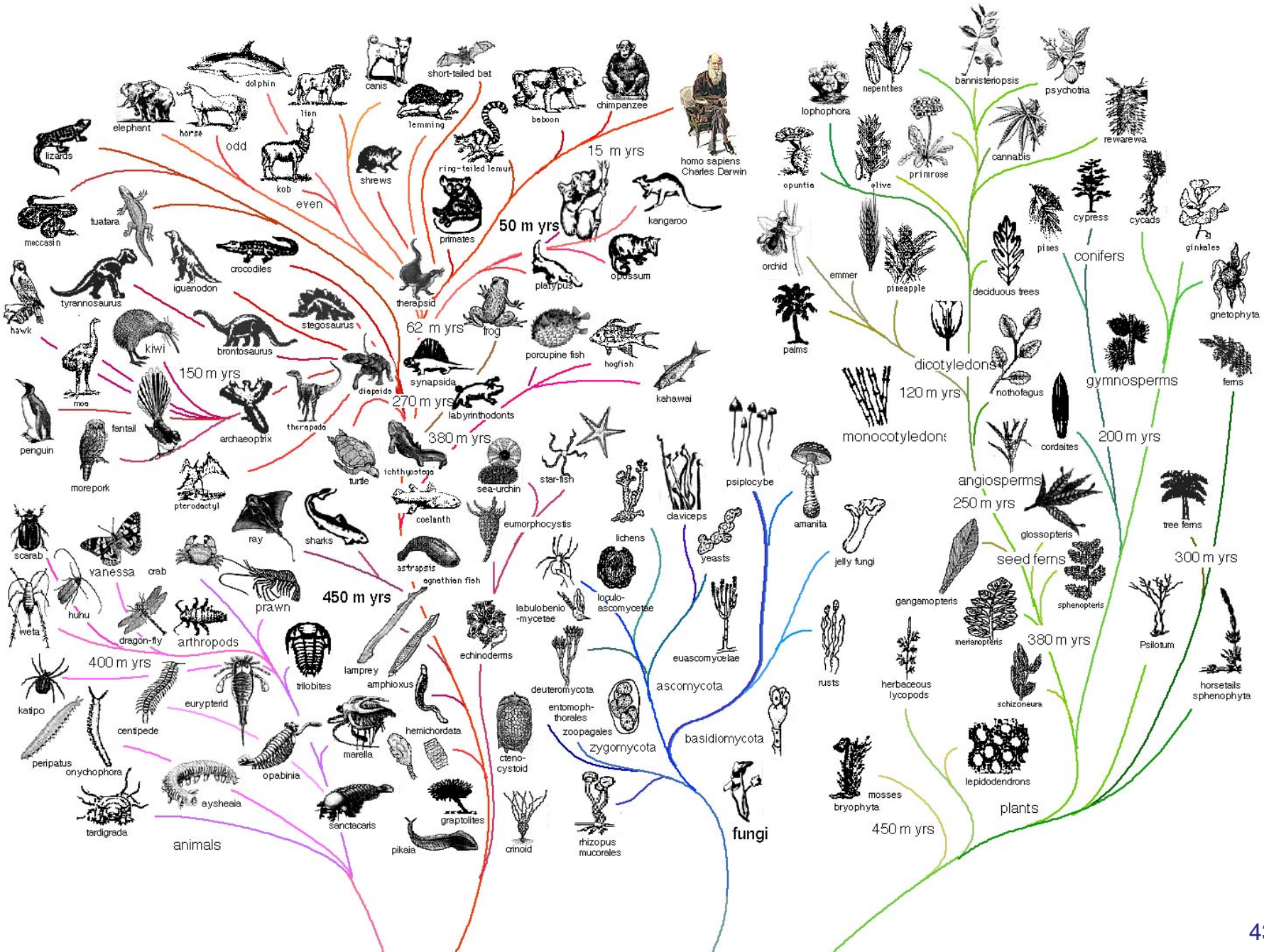
Mixture Models

Applications, Issues, Extensions

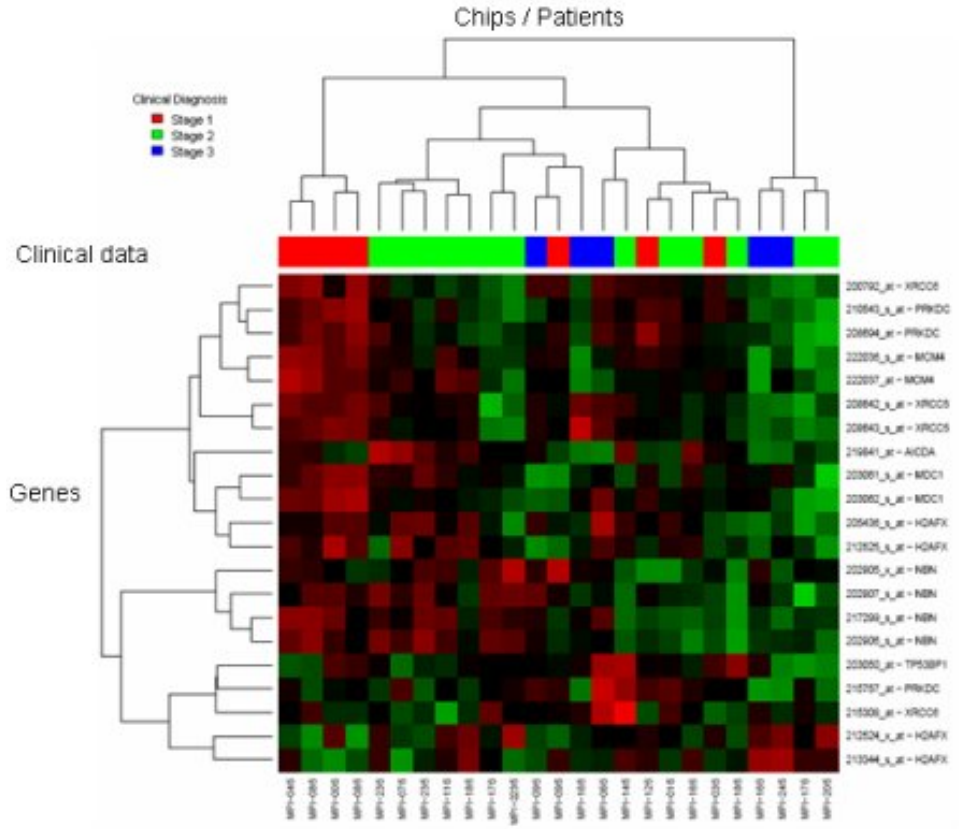
- Mixture models can be applied wherever K-means is applied.
- Mixture models are also used in density estimation tasks.
- Mixture models are strictly more powerful than K-means.
  - They can model a larger class of data sets.
  - The search space is larger and can slow down convergence of algorithm.
- Mixture models can be significantly extended within the framework of probabilistic models.
  - Extensions to the model: robust, nonparametric, non-Gaussian, mixture of probabilistic PCAs.
  - Improvements to the EM algorithm: MAP, variational Bayes, MCMC.
- Does not mean that K-means is to be replaced: K-means is computationally simpler and faster.

Outline

Hierarchical Clustering



Hierarchical Clustering



Hierarchical Clustering

Different approaches

- Top-down decimative approach.
  - Start with one big cluster.
  - Recursively split each cluster (if advantageous).
- Bottom-up agglomerative approach.
  - Start with one cluster per data point.
  - Iteratively find two clusters to merge (if advantageous).
  - Clusters found by finding pairs with maximum similarity.
- Probabilistic approaches:
  - Define a probabilistic model with a latent tree and learn the tree structure by maximum-likelihood or other techniques.
- The dominant approach is bottom-up: better search landscape, more flexible algorithms.

Hierarchical Clustering

Linkage Algorithms

- Input: data $\mathbf{x}_1, \ldots, \mathbf{x}_m$.
- Input: distance measure $d(x, y)$.
- Input: distance combination:

$$d(C, D) = f(d(x, y) : x \in C, y \in D)$$

- Initialize each data point in separate cluster:

$$C_i = \{x_i\} \text{ for } i = 1, \ldots, m$$

- For $t = 1, \ldots, m-1$:
  - Find cluster pair:

$$C, D \leftarrow \operatorname*{argmin}_{C \neq D} d(C, D)$$

  - Merge C and D: Remove C and D, add $C \cup D$.

[Duda & Hart 1973]

Hierarchical Clustering

Distance Combinations in Linkage Algorithms

- Single (or minimum) linkage:

$$d(C,D) = \min_{x \in C, y \in D} d(x,y)$$

- Complete (or maximum) linkage:

$$d(C,D) = \max_{x \in C, y \in D} d(x,y)$$

- Average linkage:

$$d(C,D) = \frac{1}{|C||D|} \sum_{x \in C, y \in D} d(x,y)$$

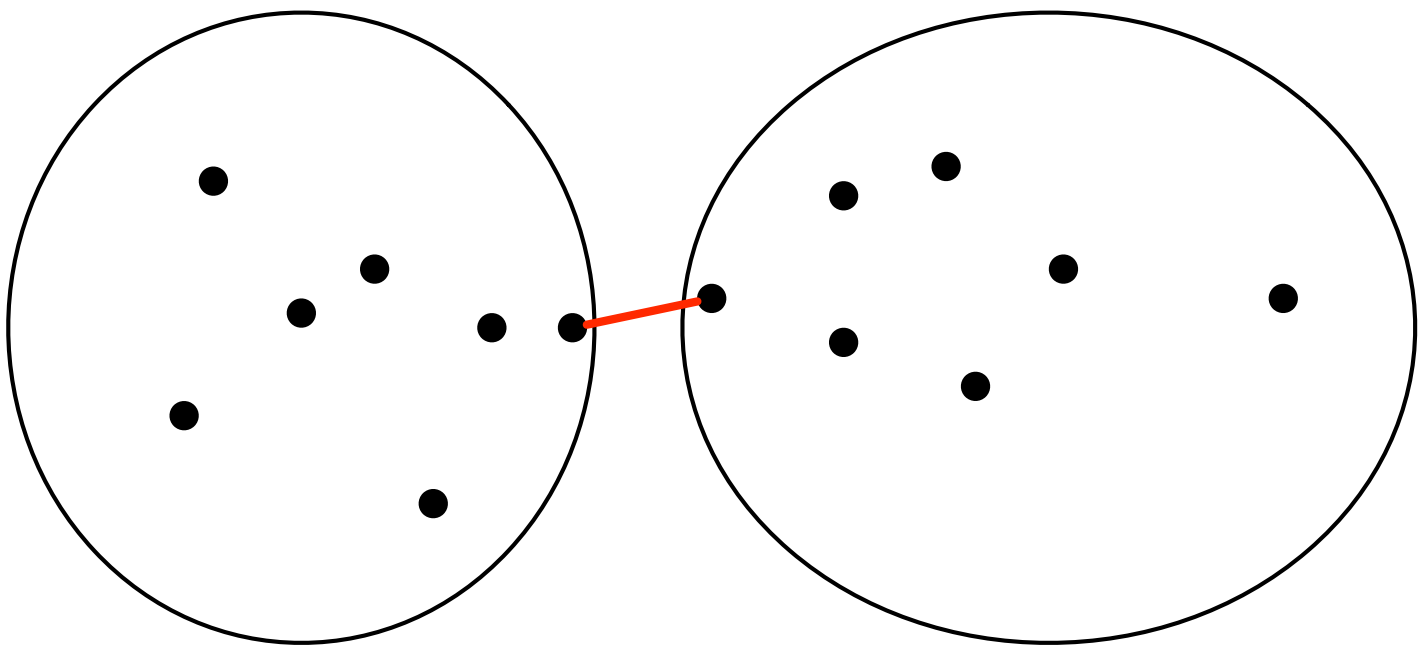
- Others: mean, centroid, ward, weighted versions...

Hierarchical Clustering

Distance Combinations in Linkage Algorithms

- Single (or minimum) linkage:

$$d(C,D) = \min_{x \in C, y \in D} d(x,y)$$

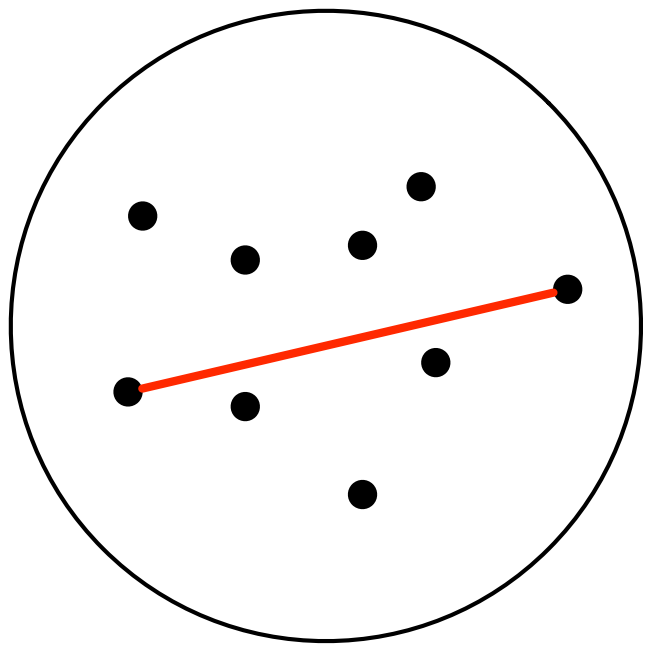


Hierarchical Clustering

Distance Combinations in Linkage Algorithms

- Complete (or maximum) linkage:

$$d(C, D) = \max_{x \in C, y \in D} d(x, y)$$



Hierarchical Clustering

Prescriptive Search Strategy

- Even given an objective function, clustering is often difficult due to its combinatorial nature—there are too many ways to cluster data.
- The problem gets even harder if we need to determine the number K of clusters in addition to the clustering.
- One way of thinking about hierarchical clustering is that it produces a *pretty good* path, from m clusters to 1 cluster, along which to search for a good K.
- This is prescriptive in that for a given K the hierarchical clustering algorithm tells you the clustering to use.

Hierarchical Clustering

Probabilistic Hierarchical Clustering

- Same framework as normal linkage algorithms.
- Use probabilistic models to define cluster distance:

$$d(C, D) = -\log \frac{p(C \cup D)}{p(C)p(D)}$$

- A common model: Gaussian

$$p(C) = \prod_{\mathbf{x} \in C} |2\pi\Psi|^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^\top \Psi^{-1}(\mathbf{x}-\mu)}$$

- The Gaussian imposes a strong constraint on how it thinks clusters should shape like.
- Clusters are merged if the merger produces a more Gaussian looking cluster.

[Friedman 2003, Heller & Ghahramani 2005]

Hierarchical Clustering

Probabilistic Hierarchical Clustering

- Different interpretation: mixture model.
- Model data set with a (standard) mixture model.
- Start with each data item x_i in its own cluster $C_i = \{x_i\}$.
- For $t = 1, \ldots, m-1$:
  - Find pair of clusters such that the likelihood of the data is maximum after merger. Equivalent to finding

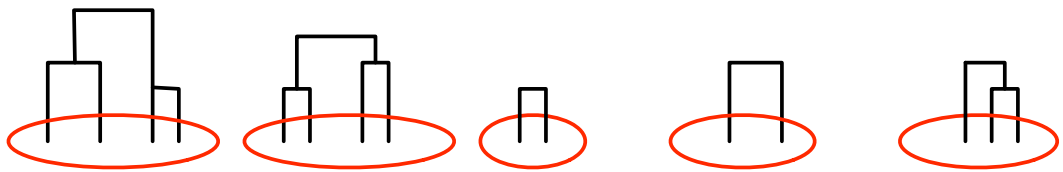
$$C, D \leftarrow \operatorname*{argmax}_{C \neq D} \log \frac{p(C \cup D)}{p(C)p(D)}$$

  - If $\log \frac{p(C \cup D)}{p(C)p(D)} > 0$ merge C and D, else stop.

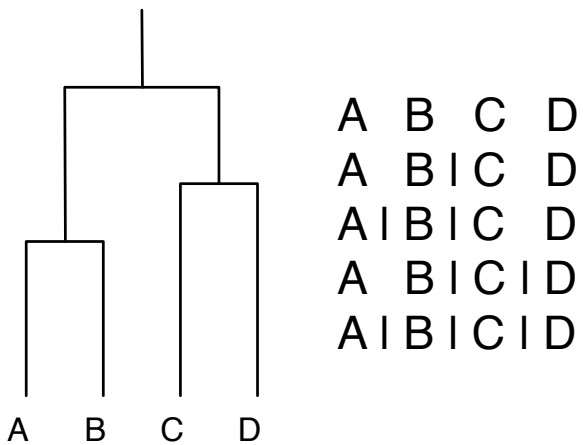
Hierarchical Clustering

Probabilistic Hierarchical Clustering

- [Friedman 2003] assumes that a partially constructed tree corresponds to a mixture model with each subtree being a mixture component.



- [Heller & Ghahramani 2005] assumes that each subtree itself corresponds to a mixture model.

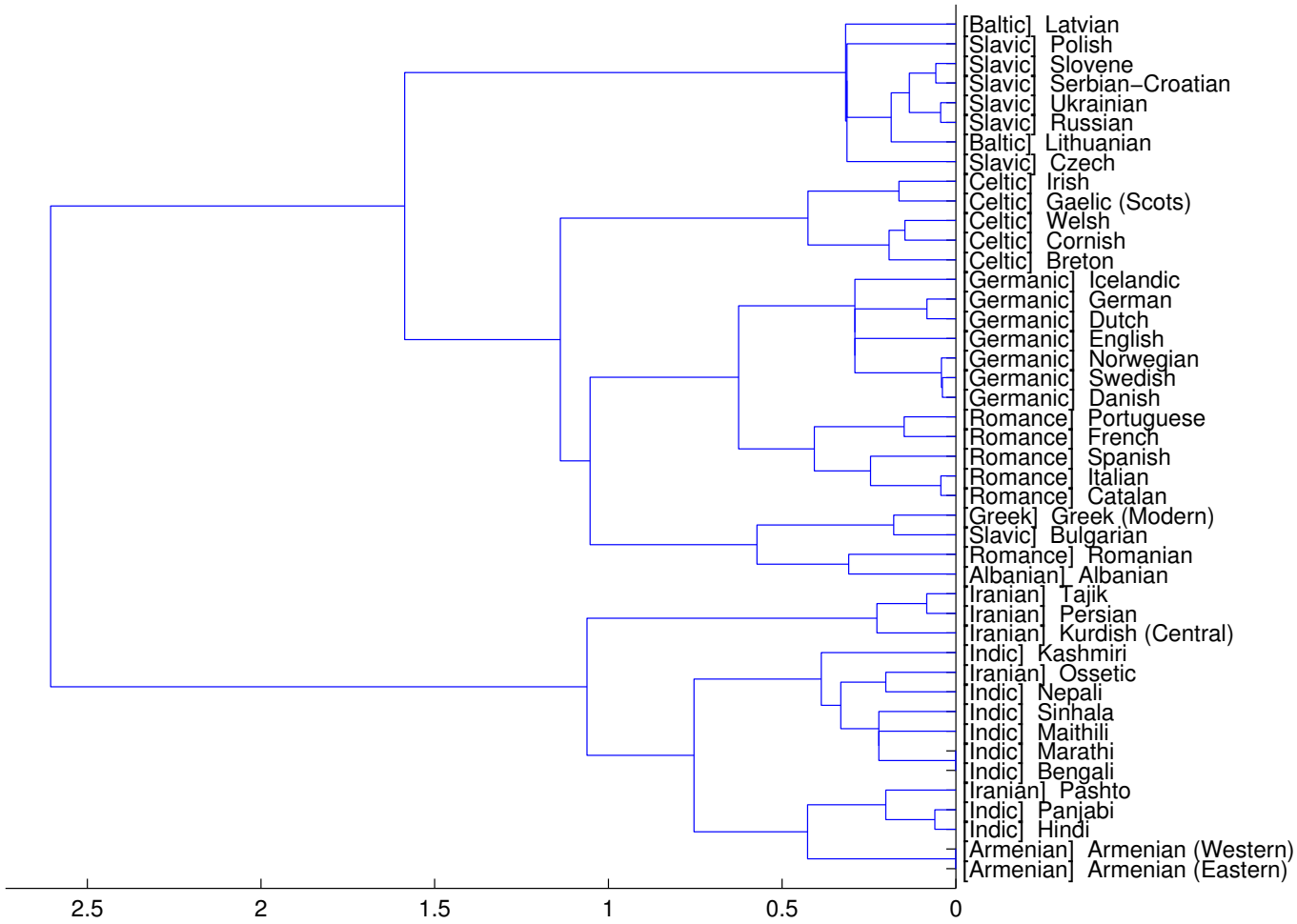


Hierarchical Clustering

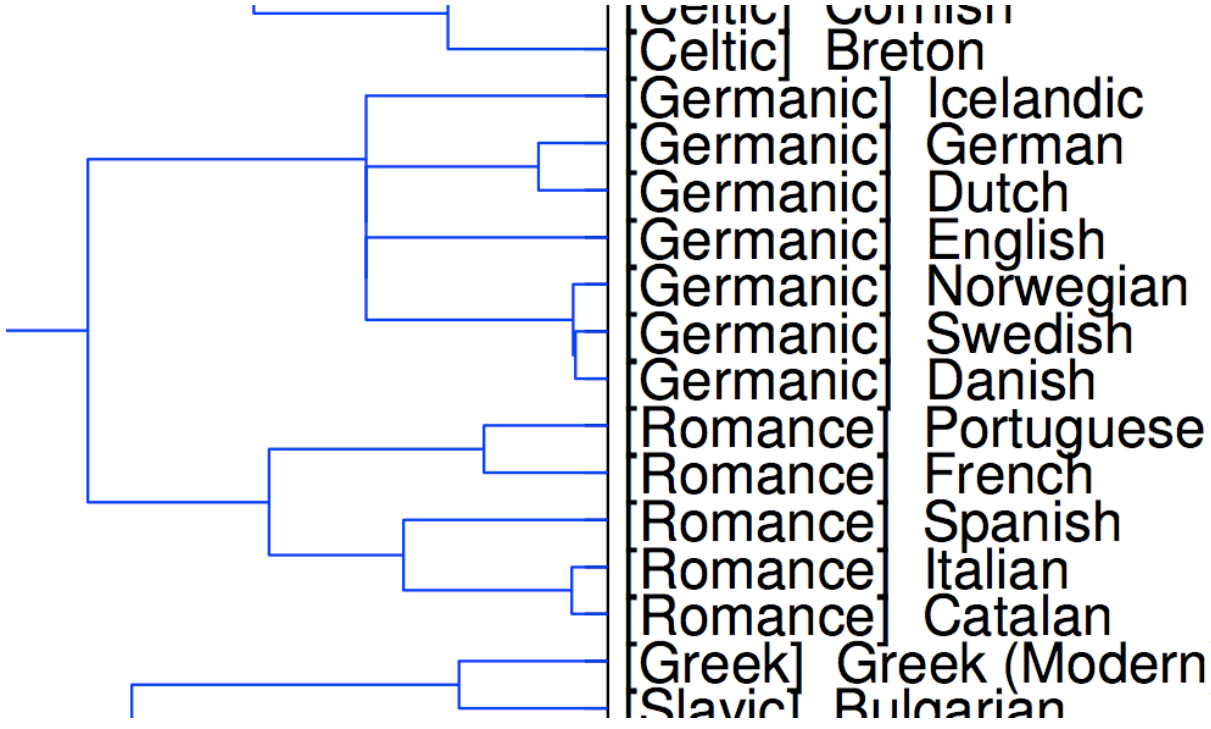
Issues, Applications

- Summarization and compression.
- Hierarchical clustering is very popular in bioinformatics.
- Prescribes a particular search strategy for mixture models: good but need not be best.
  - Can be used to initialize mixture models.
  - But computational cost is $O(m^2)$ as compared to $O(KmI)$ for mixture models, I is number of iterations.
- Hierarchical clustering is also used to discover and visualize hierarchical (tree) structure in data.
- But the algorithms we described do not optimize any objective function for quality of tree.
  - Alternatives exist: coalescents, Dirichlet diffusion trees.

Phylolinguistics



Phylolinguistics



Outline

Discussion

- A quick overview of some popular approaches to clustering.
- Applications:
  - Structure discovery, segmentation;
  - Summarization, compression;
  - Density estimation, probabilistic models.
- Dealing with high dimensions:
  - Mixtures of probabilistic PCAs, and factor analyzers.
  - Perform dimensionality reduction prior to clustering.
- Dealing with large numbers of data vectors:
  - Construct efficient data structures, e.g. KD-trees.
  - Subsample; cluster; re-cluster.
- Measuring clustering quality: indices exist, but beware!
  - Clustering is subjective; there is no right answer.
  - Clustering should be evaluated based on how much it helped achieve your goal.