Lecture 9: Classic and Modern Data Clustering

Marina Meilă
mmp@stat.washington.edu

Department of Statistics
University of Washington

November, 2014

Paradigms for clustering

Parametric clustering algorithms (K given)

- Cost based / hard clustering

Basic algorithms

- K-means clustering and the quadratic distortion
- Model based / soft clustering

Issues in parametric clustering

- Selecting K
- Outliers

Non-parametric clustering (smoothness given)

- Based on non-parametric density estimation
- Dirichlet process mixture models

Similarity based / graph clustering

- Spectral clustering
- Affinity propagation

Cluster validation

Special topics

Reading HTF: 14.3, Murphy: Ch 11.[1], 11.2.1-3, 11.3, Ch 25

What is clustering? Problem and Notation

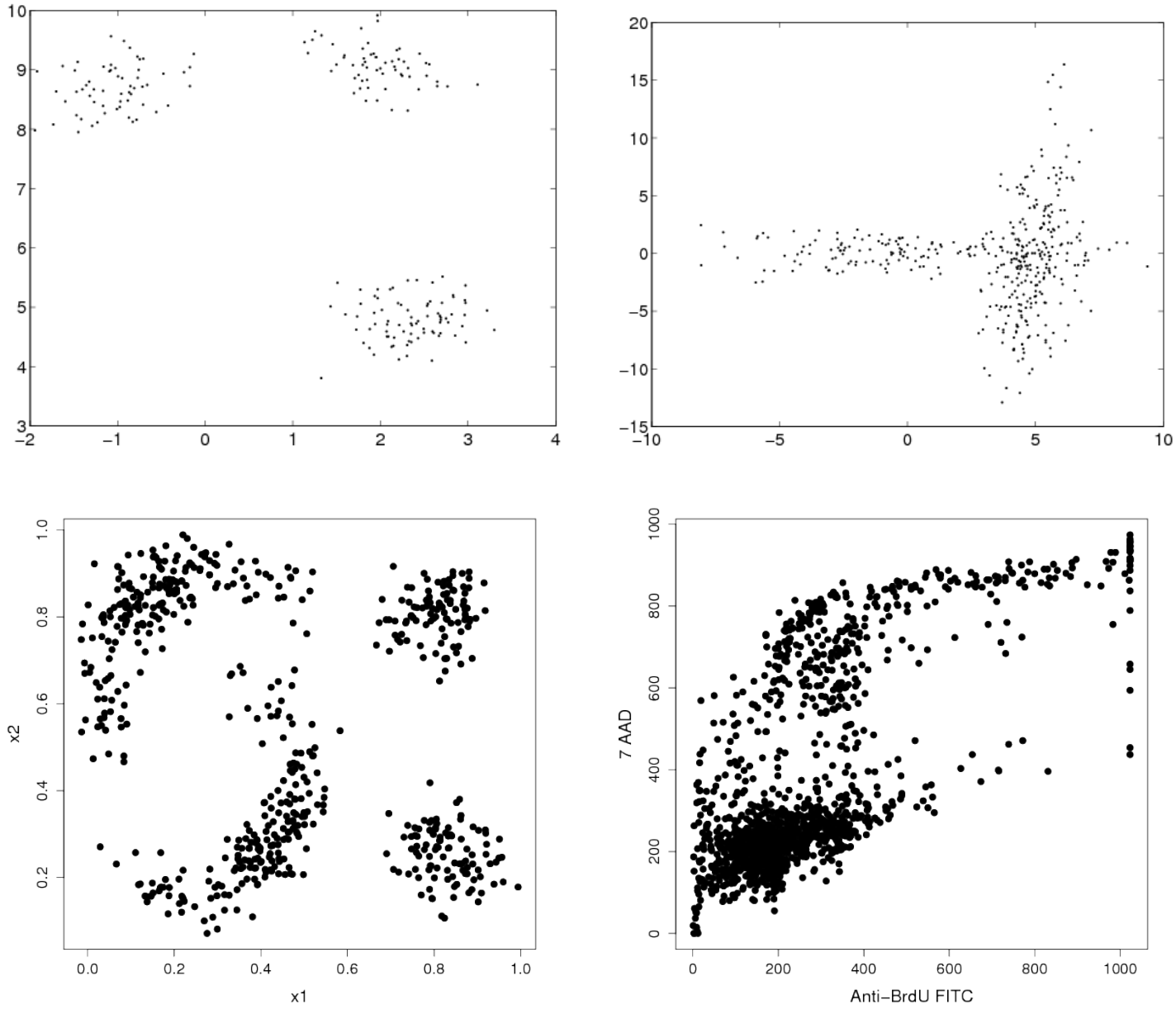
- **Informal definition** **Clustering** = Finding groups in data
- **Notation**

$\mathcal{D}$	=	$\{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n\}$ a **data set**
n	=	number of **data points**
K	=	number of **clusters** ($K << n$)
Δ	=	$\{C_1, C_2, \ldots, C_K\}$ a partition of $\mathcal{D}$ into disjoint subsets
$k(i)$	=	the **label** of point i
$\mathcal{L}(\Delta)$	=	cost (loss) of Δ (to be minimized)

- **Second informal definition** **Clustering** = given n **data points**, separate them into K **clusters**
- Hard vs. soft clusterings
  - **Hard** clustering Δ: an item belongs to only 1 cluster
  - **Soft** clustering $\gamma = \{\gamma_{ki}\}_{k=1:K}^{i=1:n}$
    γ_{ki} = the **degree of membership** of point i to cluster k

$$\sum_k \gamma_{ki} = 1 \quad \text{for all } i$$

(usually associated with a probabilistic model)



(from [Nugent and Meila, 2010])

Paradigms

Depend on type of data, type of clustering, type of cost (probabilistic or not), and constraints (about K, shape of clusters)

- Data = vectors $\{x_i\}$ in $\mathbb{R}^d$

Parametric	Cost based [hard]
(K known)	Model based [soft]
Non-parametric	Dirichlet process mixtures [soft]
(K determined	Information bottleneck [soft]
by algorithm)	Modes of distribution [hard]
	Gaussian blurring mean shift[Carreira-Perpinan, 2007] [hard]

- Data = similarities between pairs of points $[S_{ij}]_{i,j=1:n}$, $S_{ij} = S_{ji} \geq 0$ **Similarity based clustering**

Graph partitioning	spectral clustering [hard, K fixed, cost based]
	typical cuts [hard non-parametric, cost based]
Affinity propagation	[hard/soft non-parametric]

Classification vs Clustering

	Classification	**Clustering**
Cost (or Loss) $\mathcal{L}$	Expectd error	many! (probabilistic or not)
	Supervised	Unsupervised
Generalization	Performance on new data is what matters	Performance on current data is what matters
K	Known	Unknown
"Goal"	Prediction	Exploration Lots of data to explore!
Stage of field	Mature	Still young

Parametric clustering algorithms

- Cost based
  - Single linkage (min spanning tree)
  - Min diameter
    - Fastest first traversal (HS initialization)
  - K-medians
  - K-means
- Model based (cost is derived from likelihood)
  - EM algorithm
  - "Computer science" / "Probably correct" algorithms

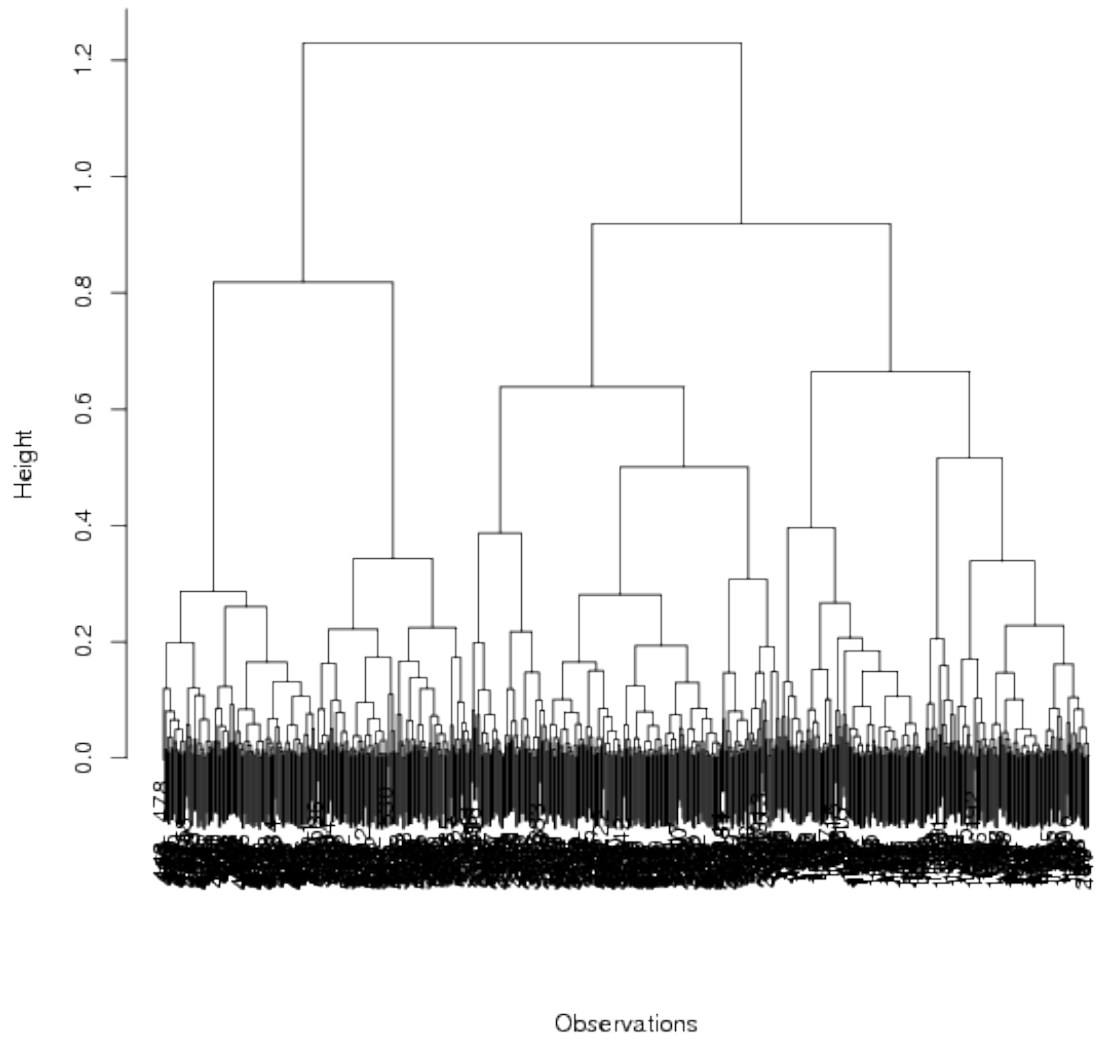
Single Linkage Clustering

Algorithm Single-Linkage

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, number clusters K

1. Construct the Minimum Spanning Tree (MST) of $\mathcal{D}$
2. Delete the largest $K-1$ edges

- **Cost** $\mathcal{L}(\Delta) = -\min_{k,k'} \text{distance}(C_k, C_{k'})$
  where $\text{distance}(A, B) = \underset{x \in A,\, y \in B}{\text{argmin}} ||x - y||$
- Running time $\mathcal{O}(n^2)$ one of the very few costs $\mathcal{L}$ that can be optimized in **polynomial** time
- Sensitive to outliers!


Height
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Observations

Minimum diameter clustering

- **Cost** $\mathcal{L}(\Delta) = \max_k \underbrace{\max_{i,j \in C_k} ||x_i - x_j||}_{\text{diameter}}$
  - Mimimize the diameter of the clusters
  - Optimizing this cost is NP-hard
- **Algorithms**
  - **Fastest First Traversal** [Hochbaum and Shmoys, 1985] – a factor 2 approximation for the min cost

  For every $\mathcal{D}$, FFT produces a Δ so that

$$\mathcal{L}^{opt} \leq \mathcal{L}(\Delta) \leq 2\mathcal{L}^{opt}$$

  - rediscovered many times

Algorithm Fastest First Traversal

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, number clusters K

defines **centers** $\mu_{1:K} \in \mathcal{D}$

(many other clustering algorithms use centers)

1. pick μ_1 at random from $\mathcal{D}$
2. for $k = 2 : K$

$$\mu_k \leftarrow \operatorname*{argmax}_{\mathcal{D}} \text{distance}(x_i, \{\mu_{1:k-1}\})$$

3. for $i = 1 : n$ (assign points to centers)

$k(i) = k$ if μ_k is the nearest center to x_i

K-medians clustering

- **Cost** $\mathcal{L}(\Delta) = \sum_k \sum i \in C_k ||x_i - \mu_k||$ with $\mu_k \in \mathcal{D}$
  - (usually) assumes centers chosen from the data points (analogy to median)

Exercise Show that in 1D $\underset{\mu}{\operatorname{argmin}} \sum_i |x_i - \mu|$ is the median of $\{x_i\}$

  - optimizing this cost is NP-hard

  - has attracted a lot of interest in theoretical CS (general from called "Facility location"

Integer Programming Formulation of K-medians

- Define $d_{ij} = ||x_i - x_j||$,
  $u_{ij} = 1$ iff point i in cluster with center x_j (0 otherwise),
  $y_j = 1$ iff point j is cluster center (0 otherwise)

$$\begin{array}{lll} \min\limits_{u,y} & \sum_{ij} d_{ij} u_{ij} & \\ \text{s.t.} & \sum_j u_{ij} = 1 & \text{point } i \text{ is in exactly 1 cluster for all } i \\ & \sum_j y_j \leq k & \text{there are at most } k \text{ clusters} \\ & u_{ij} \leq y_j & \text{point } i \text{ can only belong to a center for all } i, j \end{array}$$

Linear Programming Relaxation of K-medians

- Define d_{ij}, $y_j = 1, u_{ij}$ as before, but $y_j, u_{ij} \in [0, 1]$

$$\textbf{(LP)} \quad \begin{array}{ll} \min\limits_{u,y} & \sum_{ij} d_{ij} u_{ij} \\ \text{s.t.} & \sum_j u_{ij} = 1 \\ & \sum_j y_j \leq k \\ & u_{ij} \leq y_j \end{array}$$

Algorithm K-Medians (variant of [Bradley and Mangasarian, 2005])

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, number clusters K

1. Solve (LP)

   obtain fractionary "centers" $y_{1:n}$ and "assignments" $u_{1:n,1:n}$
2. Sample K centers $\mu_1 \dots \mu_K$ by
   - $P[\mu_k = \text{pointj}] \propto y_j$ (without replacement)
3. Assign points to centers (deterministically)

$$k(i) \;=\; \operatorname*{argmin}_k ||x_i - \mu_k||$$

- Guarantees (Agarwal)
  - **Given** tolerance ε, confidence δ, $K' = K(1 + \frac{1}{\varepsilon}) \ln \frac{n}{K}$, $\Delta_{K'}$ obtained by **K-medians** with K' centers

$$\mathcal{L}(\Delta_{K'}) \;\leq\; (1+\varepsilon)\mathcal{L}_K^{opt}$$

K-means clustering

Algorithm K-Means[Lloyd, 1982]

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, number clusters K

Initialize centers $\mu_1, \mu_2, \ldots \mu_K \in \mathbb{R}^d$ at random

Iterate until convergence

1. for $i = 1 : n$ (assign points to clusters $\Rightarrow$ new clustering)

$$k(i) = \operatorname*{argmin}_k ||x_i - \mu_k||$$

2. for $k = 1 : K$ (recalculate centers)

$$\mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i \tag{1}$$

K-means clustering

Algorithm K-Means[Lloyd, 1982]

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, number clusters K

Initialize centers $\mu_1, \mu_2, \ldots \mu_K \in \mathbb{R}^d$ at random

Iterate until convergence

1. for $i = 1 : n$ (assign points to clusters $\Rightarrow$ new clustering)

$$k(i) = \operatorname*{argmin}_k ||x_i - \mu_k||$$

2. for $k = 1 : K$ (recalculate centers)

$$\mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i \tag{1}$$

- Convergence
  - if Δ doesn't change at iteration m it will never change after that
  - convergence in finite number of steps

K-means clustering

Algorithm K-Means[Lloyd, 1982]

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, number clusters K

Initialize centers $\mu_1, \mu_2, \ldots \mu_K \in \mathbb{R}^d$ at random

Iterate until convergence

1. for $i = 1 : n$ (assign points to clusters $\Rightarrow$ new clustering)

$$k(i) = \operatorname*{argmin}_k ||x_i - \mu_k||$$

2. for $k = 1 : K$ (recalculate centers)

$$\mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i \tag{1}$$

- Convergence
  - if Δ doesn't change at iteration m it will never change after that
  - convergence in finite number of steps to local optimum of cost $\mathcal{L}$ (defined next)

K-means clustering

Algorithm K-Means[Lloyd, 1982]

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, number clusters K

Initialize centers $\mu_1, \mu_2, \ldots \mu_K \in \mathbb{R}^d$ at random

Iterate until convergence

1. for $i = 1 : n$ (assign points to clusters $\Rightarrow$ new clustering)

$$k(i) = \operatorname*{argmin}_k ||x_i - \mu_k||$$

2. for $k = 1 : K$ (recalculate centers)

$$\mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} x_i \tag{1}$$

- Convergence
  - if Δ doesn't change at iteration m it will never change after that
  - convergence in finite number of steps to local optimum of cost $\mathcal{L}$ (defined next)
  - therefore, initialization will matter

The K-means cost

$$\mathcal{L}(\Delta) = \sum_{k=1}^{K} \sum_{i \in C_k} ||x_i - \mu_k||^2 \tag{2}$$

- K-means solves a **least-squares** problem
- the cost $\mathcal{L}$ is called **quadratic distortion**

Proposition The K-means algorithm decreases $\mathcal{L}(\Delta)$ at every step.

Sketch of proof

- step 1: reassigning the labels can only decrease $\mathcal{L}$
- step 2: reassigning the centers μ_k can only decrease $\mathcal{L}$

  because μ_k as given by (1) is the solution to

$$\mu_k = \min_{\mu \in \mathbb{R}^d} \sum_{i \in C_k} ||x_i - \mu||^2 \tag{3}$$

Equivalent and similar cost functions

- The distortion can also be expressed using intracluster distances

$$\mathcal{L}(\Delta) = \sum_{k=1}^{K} \frac{1}{n_k} \sum_{i,j \in C_k} ||x_i - x_j||^2 \tag{4}$$

- **Correlation clustering** is defined as optimizing the related criterion

$$\mathcal{L}(\Delta) = \sum_{k=1}^{K} \sum_{i,j \in C_k} ||x_i - x_j||^2$$

- This cost is equivalent to the (negative) sum of (squared) intercluster distances

$$\mathcal{L}(\Delta) = -\sum_{k=1}^{K} \sum_{i \in C_k} \sum_{j \notin C_k} ||x_i - x_j||^2 + \text{constant} \tag{5}$$

Proof of (6) Replace μ_k as expressed in (1) in the expression of $\mathcal{L}$, then rearrange the terms

Proof of (5) $\sum_k \sum_{i,j \in C_k} ||x_i - x_j||^2 = \underbrace{\sum_{i=1}^{n} \sum_{j=1}^{n} ||x_i - x_j||^2}_{\text{independent of } \Delta} - \sum_k \sum_{i \in C_k} \sum_{j \notin C_k} ||x_i - x_j||^2$

The K-means cost in matrix form

- $\mathcal{L}$ as sum of squared distances to centers

$$\mathcal{L}(\Delta) = \sum_{k=1}^{K} \sum_{i \in C_k} ||x_i - \mu_k||^2$$

- $\mathcal{L}$ as sum of squared intracluster distances

$$\mathcal{L}(\Delta) = \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,j \in C_k} ||x_i - x_j||^2 \tag{6}$$

- $\mathcal{L}$ in matrix form
  Define the **assignment matrix** associated with Δ by $Z(\Delta)$
  Let $\Delta = \{C_1 = \{1, 2, 3\},\ C_2 = \{4, 5\}\}$

$$Z^{unnorm}(\Delta) = \begin{array}{c} \begin{matrix} C_1 & C_2 \end{matrix} \\ \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \end{array} \text{ point } i \qquad Z(\Delta) = \begin{array}{c} \begin{matrix} C_1 & C_2 \end{matrix} \\ \begin{bmatrix} 1/\sqrt{3} & 0 \\ 1/\sqrt{3} & 0 \\ 1/\sqrt{3} & 0 \\ 0 & 1/\sqrt{2} \\ 0 & 1/\sqrt{2} \end{bmatrix} \end{array}$$

Then Z is an orthogonal matrix (columns are orthornormal) and

$$\mathcal{L}(\Delta) = \operatorname{trace} Z^T A Z \qquad \text{with } A_{ij} = ||x_i - x_j||^2 \tag{7}$$

Proof of (7) Start from (2) and note that
$\operatorname{trace} Z^T A Z = \sum_k \sum_{i,j \in C_k} Z_{ik} Z_{jk} A_{ij} = \sum_k \sum_{i,j \in C_k} \frac{1}{|C_k|} A_{ij}$

Symmetries between costs

- K-means cost $\mathcal{L}(\Delta) = \min_{\mu_{1:K}} \sum_k \sum_{i \in C_k} ||x_i - \mu_k||^2$
- K-medians cost $\mathcal{L}(\Delta) = \min_{\mu_{1:K}} \sum_k \sum_{i \in C_k} ||x_i - \mu_k||$

- Correlation clustering cost $\mathcal{L}(\Delta) = \sum_k \sum_{i,j \in C_k} ||x_i - x_j||^2$
- min Diameter cost $\mathcal{L}^2(\Delta) = \max_k \max_{i,j \in C_k} ||x_i - x_j||^2$

Initialization of the centroids $\mu_{1:K}$

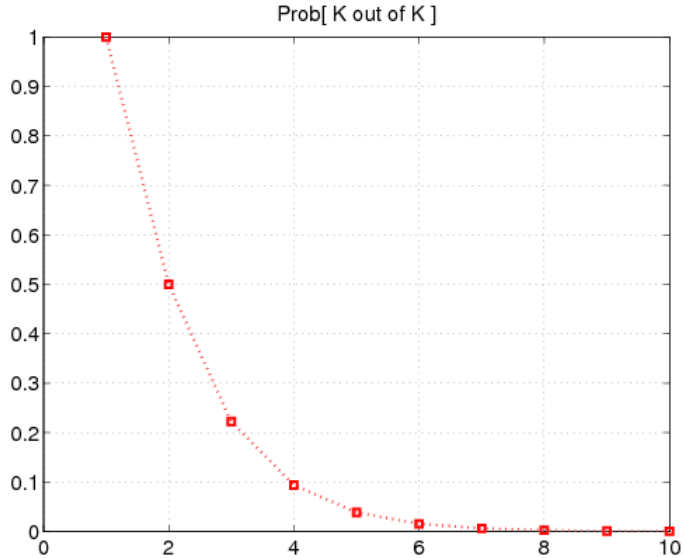
- Idea 1: start with K points at random

Initialization of the centroids $\mu_{1:K}$

- Idea 1: start with K points at random
- Idea 2: start with K data points at random

Initialization of the centroids $\mu_{1:K}$

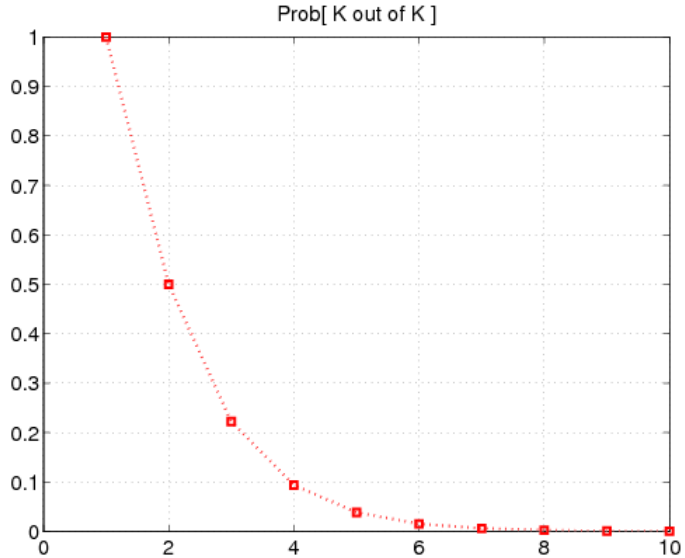
- Idea 1: start with K points at random
- Idea 2: start with K data points at random
  What's wrong with chosing K data points at random?



The probability of hitting all K clusters with K samples approaches 0 when $K > 5$

Initialization of the centroids $\mu_{1:K}$

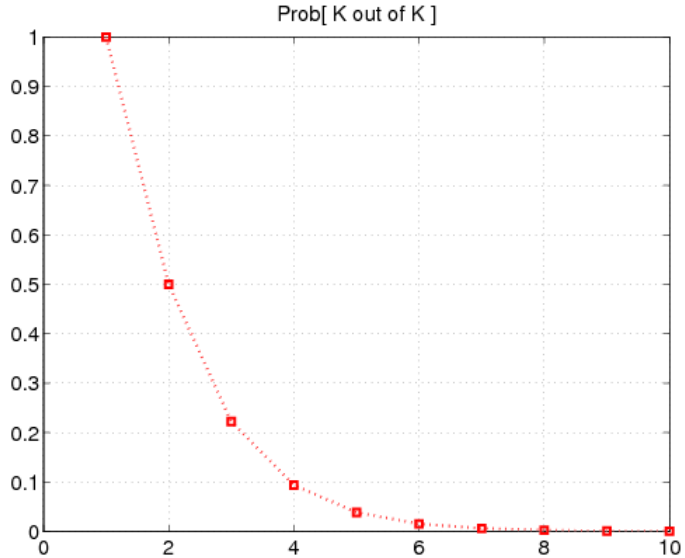
- Idea 1: start with K points at random
- Idea 2: start with K data points at random
  What's wrong with chosing K data points at random?



  The probability of hitting all K clusters with K samples approaches 0 when $K > 5$
- Idea 3: start with K data points using Fastest First Traversal [] (greedy simple approach to spread out centers)

Initialization of the centroids $\mu_{1:K}$

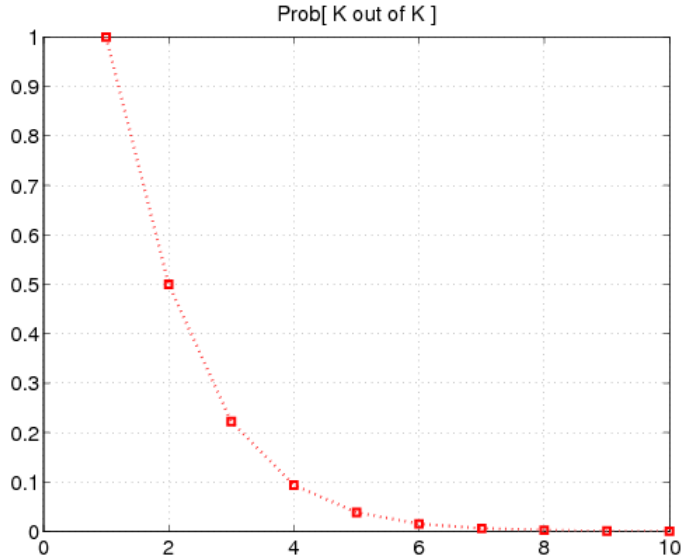
- Idea 1: start with K points at random
- Idea 2: start with K data points at random
  What's wrong with chosing K data points at random?



  The probability of hitting all K clusters with K samples approaches 0 when $K > 5$
- Idea 3: start with K data points using Fastest First Traversal [] (greedy simple approach to spread out centers)
- Idea 4: **k-means++** [] (randomized, theoretically backed approach to spread out centers)

Initialization of the centroids $\mu_{1:K}$

- Idea 1: start with K points at random
- Idea 2: start with K data points at random
  What's wrong with chosing K data points at random?



  The probability of hitting all K clusters with K samples approaches 0 when $K > 5$
- Idea 3: start with K data points using Fastest First Traversal [] (greedy simple approach to spread out centers)
- Idea 4: **k-means++** [] (randomized, theoretically backed approach to spread out centers)
- Idea 5: **"K-logK" Initialization** (start with enough centers to hit all clusters, then prune down to K)
  For EM Algorithm [], for K-means [Bubeck et al., 2009]

The "K-logK" initialization

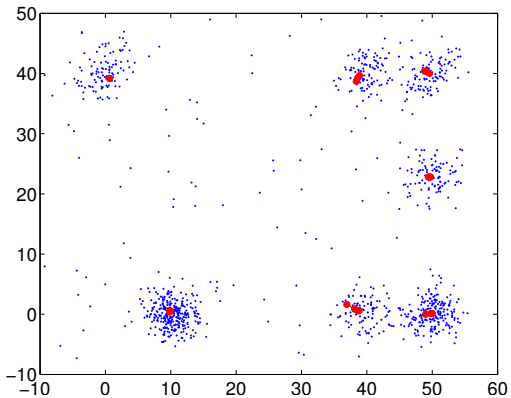
The K-logK Initialization (see also [Bubeck et al., 2009])

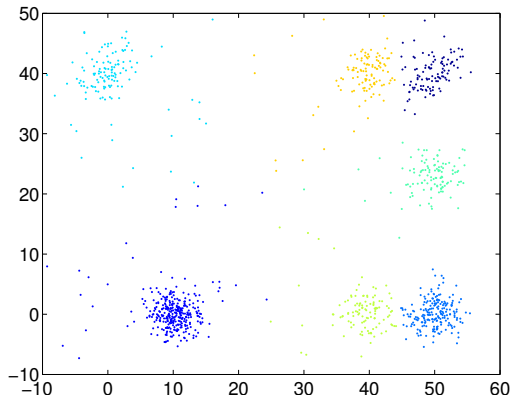
1. pick $\mu^0_{1:K'}$ at random from data set, where $K' = O(K \log K)$
   (this assures that each cluster has at least 1 center w.h.p)
2. run 1 step of K-means
3. remove all centers μ^0_k that have few points, e.g $|C_k| < \frac{n}{eK'}$
4. from the remaining centers select K centers by **Fastest First Traversal**
   - 4.1 pick μ_1 at random from the remaining $\{\mu^0_{1:K'}\}$
   - 4.2 for $k = 2 : K$, $\mu_k \leftarrow \underset{\mu^0_{k'}}{\operatorname{argmax}} \min_{j=1:k-1} ||\mu^0_{k'} - \mu_j||$, i.e next μ_k is furthest away from the already chosen centers
5. continue with the standard **K-means** algorithm

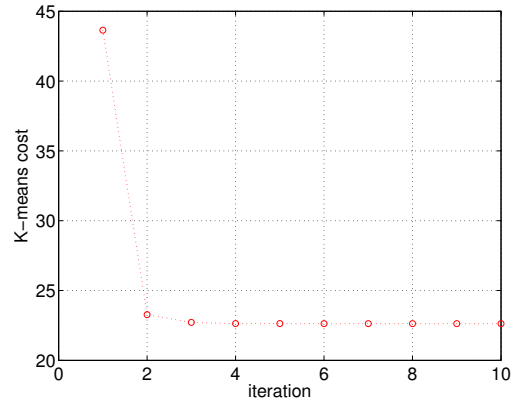
K-means clustering with K-logK Initialization

Example using a mixture of 7 Normal distributions with 100 outliers sampled uniformly

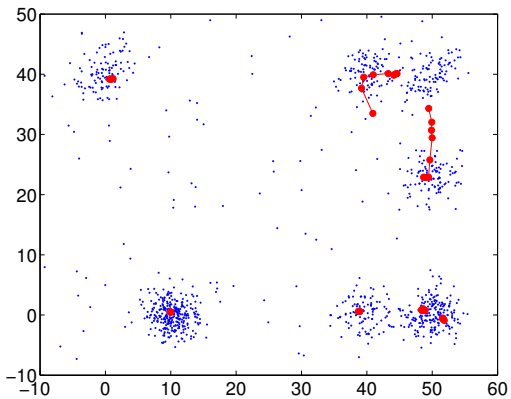
K-LOGK $K = 7,\ T = 100,\ n = 1100,\ c = 1$

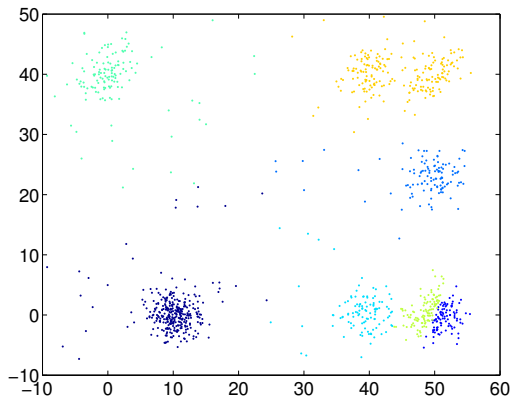


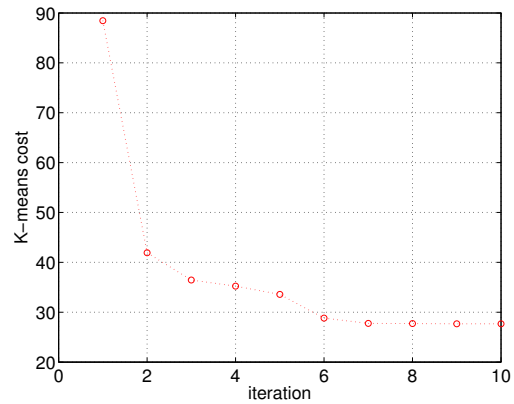




NAIVE $K = 7\ T = 100,\ n = 1100$







Coresets approach to K-medians and K-means

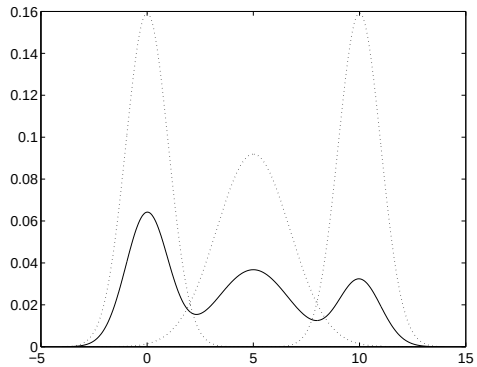
- A weighted subset of $\mathcal{D}$ is a (K, ε) **coreset** iff for any $\mu_{1:K}$,

$$|\mathcal{L}(\mu_{1:K}, A) - \mathcal{L}(\mu_{1:K}; \mathcal{D})| \leq \varepsilon \mathcal{L}(\mu_{1:K}; \mathcal{D})$$

- Note that the size of A is not K
- Finding a coreset (fast) lets use find fast algorithms for clustering a large $\mathcal{D}$
  - "fast" $=$ linear in n, exponential in ε^{-d}, polynomial in K
- **Theorem**[Har-Peled and Mazumdar, 2004],Theorem 5.7
  One can compute an $(1+\varepsilon)$-approximate K-median of a set of n points in time $\mathcal{O}(n + K^5 \log^9 n + gK^2 \log^5 n)$ where $g = e^{[C/\varepsilon \log(1+1/\varepsilon)]^{d-1}}$ (where d is the dimension of the data)
- **Theorem**[Har-Peled and Mazumdar, 2004],Theorem 6.5
  One can compute an $(1+\varepsilon)$-approximate K-means of a set of n points in time $\mathcal{O}(n + K^5 \log^9 n + K^{K+2} \varepsilon^{-(2d+1)} \log^{K+1} n \log^K \frac{1}{\varepsilon})$.

Model based clustering: Mixture models

Mixture in 1D

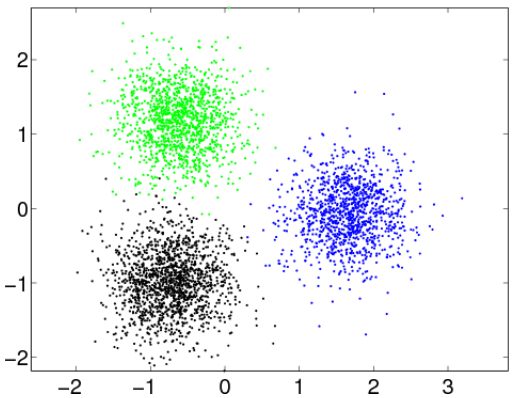


- The **mixture density**

$$f(x) = \sum_{k=1}^{K} \pi_k f_k(x)$$

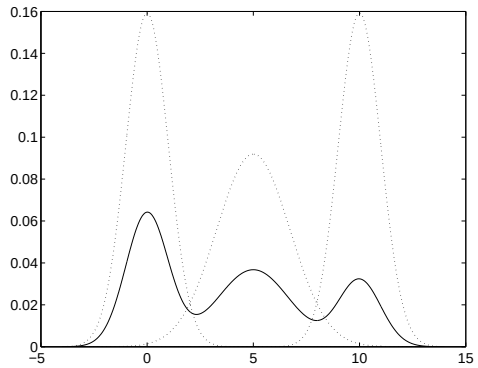
- $f_k(x)$ = the **components** of the mixture
  - each is a density
  - f called **mixture of Gaussians** if $f_k = Normal_{\mu_k, \Sigma_k}$
- π_k = the **mixing proportions**, $\sum_k = 1^K \pi_k = 1, \quad \pi_k \geq 0.$
- **model parameters** $\theta = (\pi_{1:K}, \mu_{1:K}, \Sigma_{1:K})$

Mixture in 2D



Model based clustering: Mixture models

Mixture in 1D

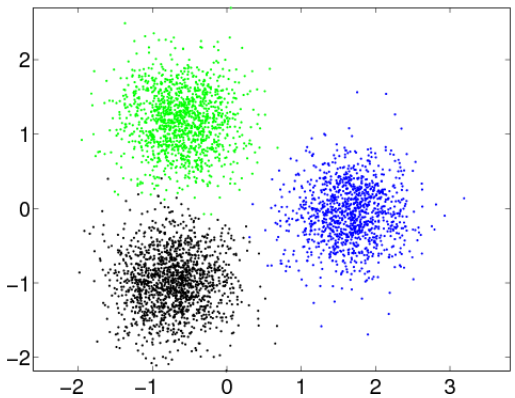


- The **mixture density**

$$f(x) = \sum_{k=1}^{K} \pi_k f_k(x)$$

- $f_k(x)$ = the **components** of the mixture
  - each is a density
  - f called **mixture of Gaussians** if $f_k = Normal_{\mu_k, \Sigma_k}$
- π_k = the **mixing proportions**, $\sum_k = 1^K \pi_k = 1, \quad \pi_k \geq 0.$
- **model parameters** $\theta = (\pi_{1:K}, \mu_{1:K}, \Sigma_{1:K})$

Mixture in 2D



- The **degree of membership** of point i to cluster k

$$\gamma_{ki} \stackrel{\text{def}}{=} P[x_i \in C_k] = \frac{\pi_k f_k(x)}{f(x)} \text{ for } i = 1:n, k = 1:K \tag{8}$$

- depends on x_i and on the model parameters

Criterion for clustering: Max likelihood

- denote $\theta = (\pi_{1:K}, \mu_{1:K}, \Sigma_{1:K})$ (the parameters of the mixture model)
- Define **likelihood** $P[\mathcal{D}|\theta] = \prod_{i=1}^{n} f(x_i)$
- Typically, we use the **log likelihood**

$$l(\theta) = \ln \prod_{i=1}^{n} f(x_i) = \sum_{i=1}^{n} \ln \sum_{k} \pi_k f_k(x_i) \tag{9}$$

- denote $\theta^{ML} = \operatorname{argmax}_{\theta} l(\theta)$
- θ^{ML} determines a soft clustering γ by (8)
- a soft clustering γ determines a θ (see later)
- Therefore we can write

$$\mathcal{L}(\gamma) = -l(\theta(\gamma))$$

Algorithms for model-based clustering

Maximize the (log-)likelihood w.r.t θ

- directly - (e.g by gradient ascent in θ)
- by the EM algorithm (very popular!)
- indirectly, w.h.p. by "computer science" algorithms

w.h.p = with high probability (over data sets)

The Expectation-Maximization (EM) Algorithm

Algorithm Expectation-Maximization (EM)

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, number clusters K

Initialize parameters $\pi_{1:K} \in \mathbb{R}$, $\mu_{1:K} \in \mathbb{R}^d$, $\Sigma_{1:K} \in \mathbb{R}^{d \times d}$ at random[1]

Iterate until convergence

E step (Optimize clustering) for $i = 1 : n$, $k = 1 : K$

$$\gamma_{ki} = \frac{\pi_k f_k(x)}{f(x)}$$

M step (Optimize parameters) set $\Gamma_k = \sum_{i=1}^n \gamma_{ki}$, $k = 1 : K$ (number of points in cluster k)

$$\pi_k = \frac{\Gamma_k}{n}, \quad k = 1 : K$$

$$\mu_k = \sum_{i=1}^n \frac{\gamma_{ki}}{\Gamma_k} x_i$$

$$\Sigma_k = \frac{\sum_{i=1}^n \gamma_{ki}(x_i - \mu_k)(x_i - \mu_k)^T}{\Gamma_k}$$

- $\pi_{1:K}, \mu_{1:K}, \Sigma_{1:K}$ are the maximizers of $l_c(\theta)$ in (13)
- $\sum_k \Gamma_k = n$

[1] Σ_k need to be symmetric, positive definite matrices

The EM Algorithm – Motivation

- Define the **indicator variables**

$$z_{ik} = \begin{cases} 1 & \text{if } i \in C_k \\ 0 & \text{if } i \notin C_k \end{cases} \tag{10}$$

  denote $\bar{z} = \{z_{ki}\}_{k=1:K}^{i=1:n}$

- Define the **complete log-likelihood**

$$l_c(\theta, \bar{z}) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ki} \ln \pi_k f_k(x_i) \tag{11}$$

- $E[z_{ki}] = \gamma_{ki}$
- Then

$$E[l_c(\theta, \bar{z})] = \sum_{i=1}^{n} \sum_{k=1}^{K} E[z_{ki}][\ln \pi_k + \ln f_k(x_i)] \tag{12}$$

$$= \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ki} \ln \pi_k + \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ki} \ln f_k(x_i)] \tag{13}$$

- If θ known, γ_{ki} can be obtained by (8) **(Expectation)**
- If γ_{ki} known, π_k, μ_k, Σ_k can be obtained by separately maximizing the terms of $E[l_c]$ **(Maximization)**

Brief analysis of EM

$$Q(\theta, \gamma) = \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{ki} \ln \underbrace{\pi_k f_k(x_i)}_{\theta}$$

- each step of EM increases $Q(\theta, \gamma)$
- Q converges to a local maximum
- at every local maxi of Q, $\theta \leftrightarrow \gamma$ are fixed point
- $Q(\theta^*, \gamma^*)$ local max for $Q \Rightarrow$ $l(\theta^*)$ local max for $l(\theta)$
- under certain regularity conditions $\theta \longrightarrow \theta^{ML}$ [McLachlan and Krishnan, 1997]
- the E and M steps can be seen as projections [Neal and Hinton, 1998]

- Exact maximization in **M step** is not essential.
  Sufficient to increase Q.
  This is called **Generalized EM**

Probablistic alternate projection view of EM[Neal and Hinton, 1998]

- let z_i = which gaussian generated i? (random variable), $X = (x_{1:n})$, $Z = (z_{1:n})$
- Redefine Q

$$Q(\tilde{P}, \theta) \;=\; \mathcal{L}(\theta) - KL(\tilde{P}||P(Z|X,\theta)$$

where $P(X, Z|\theta) = \prod_i \prod_k P[z_i = k]P[x_i|\theta_k]$
$\tilde{P}(Z)$ is any distribution over Z,
$KL(P(w)||Q(w)) = \sum_w P(w) \ln \frac{P(w)}{Q(w)}$ the **Kullbach-Leibler divergence**

Then,

- **E step** $\max_{\tilde{P}} Q \Leftrightarrow \; KL(\tilde{P}||P(Z|X,\theta)$
- **M step** $\max_{\theta} Q \Leftrightarrow \; KL(P(X|Z,\theta^{old})||P(X|\theta))$

- Interpretation: KL is "distance", "shortest distance" = projection

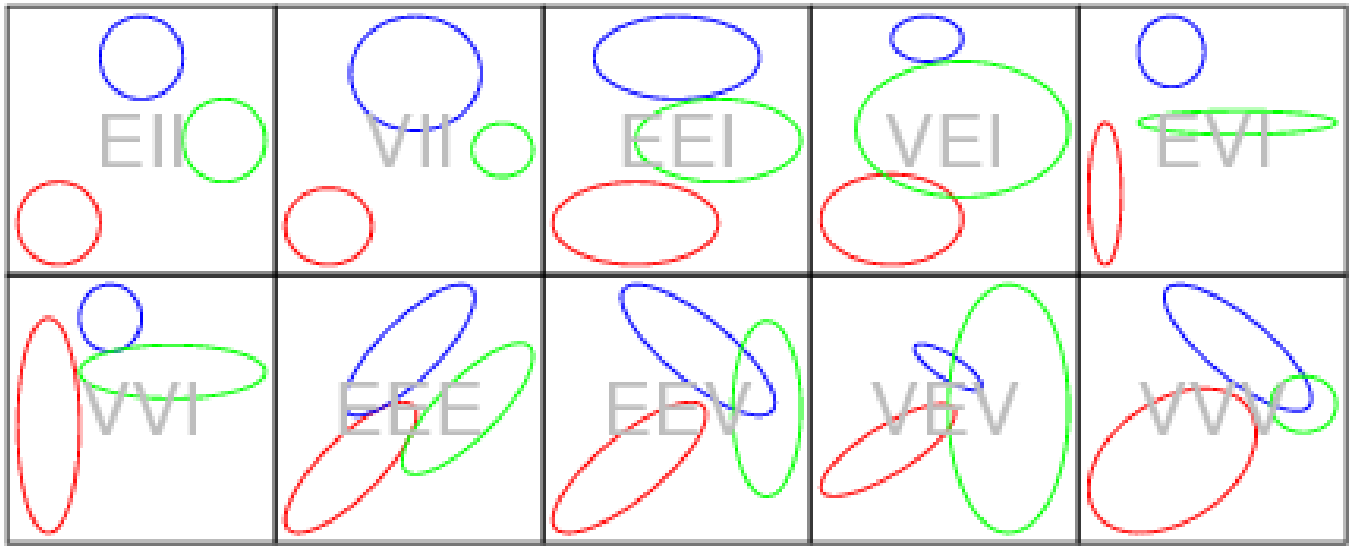
The M step in special cases

- Note that the expressions for μ_k, Σ_k = expressions for μ, Σ in the normal distribution, with data points x_i weighted by $\frac{\gamma_{ki}}{\Gamma_k}$

	M step
general case	$\Sigma_k = \sum_{i=1}^{n} \frac{\gamma_{ki}}{\Gamma_k}(x_i - \mu_k)(x_i - \mu_k)^T$
$\Sigma_k = \Sigma$ "same shape & size" clusters	$\Sigma \leftarrow \frac{\sum_{i=1}^{n}\sum_{k=1}^{K} \gamma_{ki}(x_i-\mu_k)(x_i-\mu_k)^T}{n}$
$\Sigma_k = \sigma_k^2 I_d$ "round" clusters	$\sigma_k^2 \leftarrow \frac{\sum_{i=1}^{n} \gamma_{ki}\|\|x_i-\mu_k\|\|^2}{d\Gamma_k}$
$\Sigma_k = \sigma^2 I_d$ "round, same size" clusters	$\sigma^2 \leftarrow \frac{\sum_{i=1}^{n}\sum_{k=1}^{K} \gamma_{ki}\|\|x_i-\mu_k\|\|^2}{nd}$

Exercise Prove the formulas above

- Note also that **K-means** is **EM** with $\Sigma_k = \sigma^2 I_d$, $\sigma^2 \to 0$ Exercise Prove it



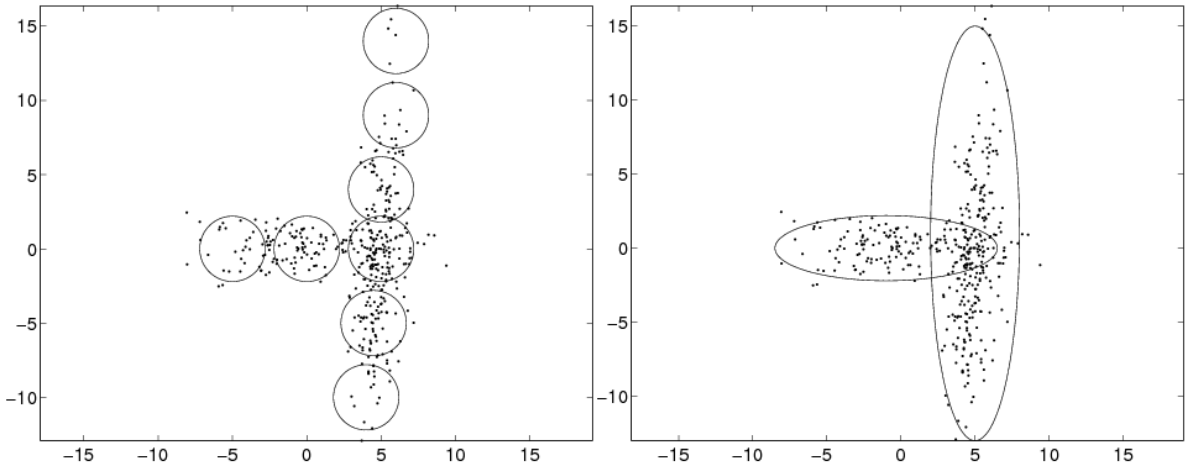
More special cases [Banfield and Raftery, 1993] introduce the following description for a covariance matrice in terms of *volume, shape, alignment with axes* (=determinant, trace, e-vectors). The letters below mean: I=unitary (shape, axes), E=equal (for all k), V=unequal

- EII: equal volume, round shape (spherical covariance)
- VII: varying volume, round shape (spherical covariance)
- EEI: equal volume, equal shape, axis parallel orientation (diagonal covariance)
- VEI: varying volume, equal shape, axis parallel orientation (diagonal covariance)
- EVI: equal volume, varying shape, axis parallel orientation (diagonal covariance)
- VVI: varying volume, varying shape, equal orientation (diagonal covariance)
- EEE: equal volume, equal shape, equal orientation (ellipsoidal covariance)
- EEV: equal volume, equal shape, varying orientation (ellipsoidal covariance)
- VEV: varying volume, equal shape, varying orientation (ellipsoidal covariance)
- VVV: varying volume, varying shape, varying orientation (ellipsoidal covariance)

(from [Nugent and Meila, 2010])

EM versus K-means

- Alternates between cluster assignments and parameter estimation
- Cluster assignments γ_{ki} are probabilistic
- Cluster parametrization more flexible



- Converges to local optimum of **log-likelihood**

  Initialization recommended by K-logK method []

- **Modern algorithms with guarantees** (for e.g. mixtures of Gaussians)
  - Random projections
  - Projection on principal subspace [Vempala and Wang, 2004]
  - Two step EM (=K-logK initialization + one more EM iteration) []

"Computer science" algorithms for mixture models

- Assume clusters well-separated **(S)**
  - e.g $||\mu_k - \mu_l|| \geq C \max(\sigma_k, \sigma_l)$
  - with $\sigma_k^2 = \max \text{ eigenvalue}(\Sigma_k)$
- true distribution is mixture
  - of Gaussians
  - of **log-concave** f_k's (i.e. $\ln f_k$ is concave function)
- then, w.h.p. (n, K, d, C)
  - we can label all data points correctly
  - $\Rightarrow$ we can find good estimate for θ

Even with **(S)** this is not an easy task in high dimensions

Because $f_k(\mu_k) \to 0$ in high dimensions (i.e there are few points from Gaussian k near μ_k)

The Vempala-Wang algorithm[Vempala and Wang, 2004]

Idea

Let $\mathcal{H} = \mathrm{span}(\mu_{1:K})$

Projecting data on $\mathcal{H}$

- $\approx$ preserves $||x_i - x_j||$ if $k(i) \neq k(j)$
- $\approx$ reduces $||x_i - x_j||$ if $k(i) = k(j)$
- density at μ_k increases

(Proved by Vempala & Wang, 2004[Vempala and Wang, 2004]) $\mathcal{H} \approx K$-th principal subspace of data

Algorithm Vempala-Wang (sketch)

1. Project points $\{x_i\} \in \mathbb{R}^d$ on $K-1$-th principal subspace $\Rightarrow$ $\{y_i\} \in \mathbb{R}^K$
2. do distance-based "harvesting" of clusters in $\{y_i\}$

Other "CS" algorithms I

- [Dasgupta, 2000] round, equal sized Gaussian, random projection
- [Arora and Kannan, 2001] arbitrary shaped Gaussian, distances
- [Achlioptas and McSherry, 2005] log-concave, principal subspace projection

Example Theorem (Achlioptas & McSherry, 2005) If data come from K Gaussians, $n >> K(d + \log K)/\pi_{min}$, and

$$||\mu_k - \mu_l|| \geq 4\sigma_k\sqrt{1/\pi_k + 1/\pi_l} + 4\sigma_k\sqrt{K \log nK + K^2}$$

then, w.h.p. $1 - \delta(d, K, n)$, their algorithm finds true labels

Good

- theoretical guarantees
- no local optima
- suggest heuritics for EM K-means
  - project data on principal subspace (when $d >> K$)

But

- strong assuptions: large separation (unrealistic), concentration of f_k's (or f_k known), K known
- try to find perfect solution (too ambitious)

A fundamental result

The Johnson-Lindenstrauss Lemma For any $\varepsilon \in (0, 1]$ and any integer n, let d' be a positive integer such that $d' \geq 4(\varepsilon^2/2 - \varepsilon^3/3)^{-1} \ln n$. Then for any set $\mathcal{D}$ of n points in $\mathbb{R}^d$, there is a map $f : \mathbb{R}^d \to \mathbb{R}^{d'}$ such that for all $u, v \in V$,

$$(1 - \varepsilon)||u - v||^2 \leq ||f(u) - f(v)||^2 \leq (1 + \varepsilon)||u - v||^2 \tag{14}$$

Furthermore, this map can be found in randomized polynomial time.

- note that the **embedding dimension** d' does not depend on the original dimension d, but depends on n, ε
- [Dasgupta and Gupta, 2002] show that: the mapping f is linear and that w.p. $1 - \frac{1}{n}$ a random projection (rescaled) has this property
- their proof is elementary Projecting a fixed vector v on a a random subspace is the same as projecting a random vector v on a fixed subspace. Assume $v = [v_1, \ldots v_d]$ with $v \sim$ i.i.d. and let $\tilde{v}$ = projection of v on axes $1 : d'$. Then $E[||\tilde{v}||^2 = d' E[v_j^2] = \frac{d'}{d} E[||v||^2]$. The next step is to show that the variance of $||\tilde{v}||^2$ is very small when d' is sufficiently large.

A two-step EM algorithm [Dasgupta and Schulman, 2007]

Assumes K spherical gaussians, separation $||\mu_k^{true} - \mu_{k'}^{true} \geq C\sqrt{d}\sigma_k$

1. Pick $K' = \mathcal{O}(K \ln K)$ centers μ_k^0 at random from the data
2. Set $\sigma_k^0 = \frac{d}{2} \min_{k \neq k'} ||\mu_k^0 - \mu_{k'}^0||^2$, $\pi_k^0 = 1/K'$
3. Run one E step and one M step $\Longrightarrow \{\pi_k^1, \mu_k^1, \sigma_k^1\}_{k=1:K'}$
4. Compute "distances" $d(\mu_k^1, \mu_{k'}^1) = \frac{||\mu_k^1 - \mu_{k'}^1||}{\sigma_k^1 - \sigma_{k'}^1}$
5. Prune all clusters with $\pi_k^1 \leq 1/4K'$
6. Run Fastest First Traversal with distances $d(\mu_k^1, \mu_{k'}^1)$ to select K of the remaining centers. Set $\pi_k^1 = 1/K$.
7. Run one E step and one M step $\Longrightarrow \{\pi_k^2, \mu_k^2, \sigma_k^2\}_{k=1:K}$

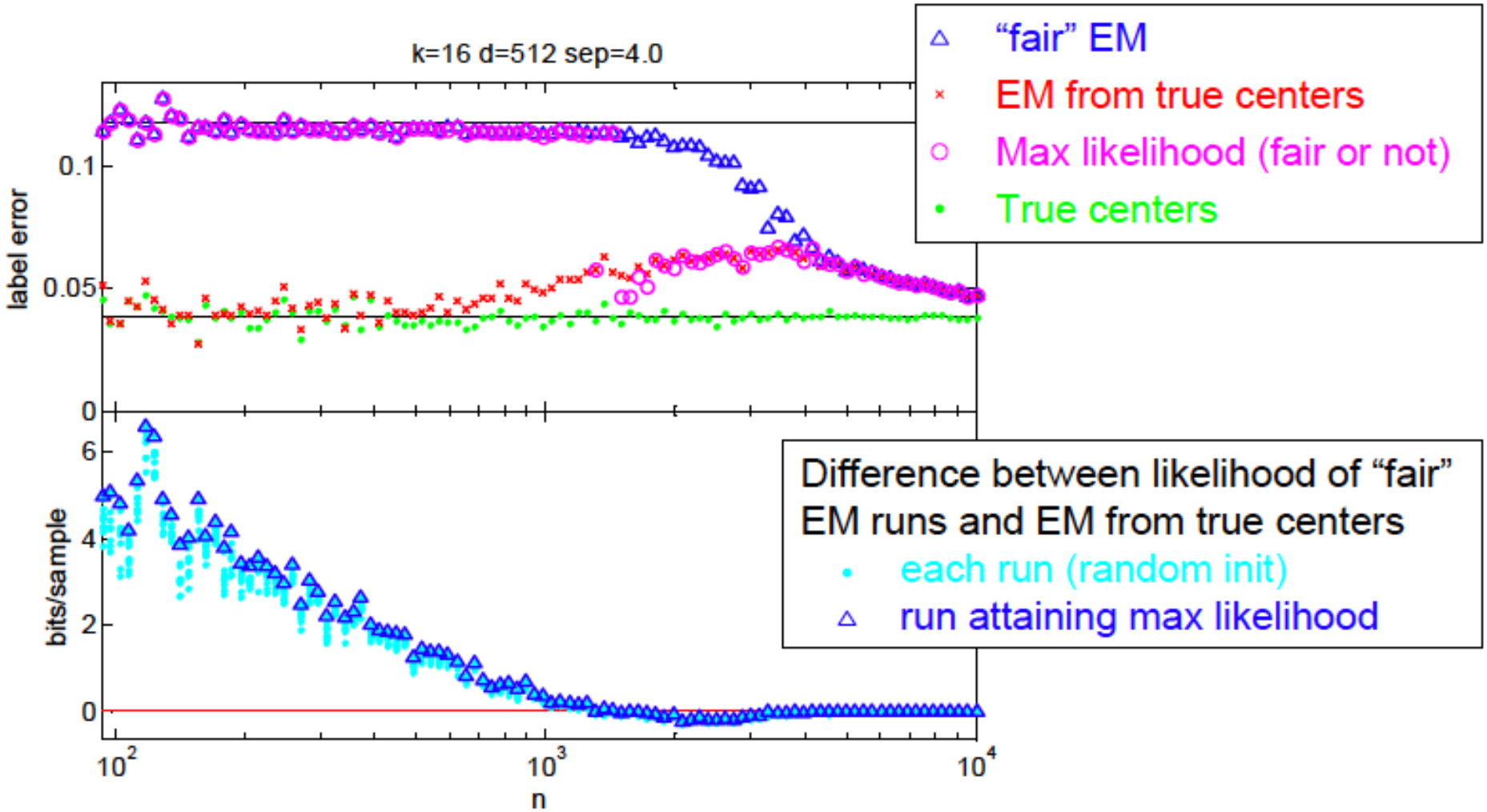
Theorem For any $\delta, \varepsilon > 0$ if d large, n large enough, separation $C \geq d^{1/4}$ the **Two step EM** algorithm obtains centers μ_k so that

$$||\mu_k - \mu_k^{true}|| \leq ||\text{mean}(C_k^{true}) - \mu_k^{true}|| + \varepsilon\sigma_k\sqrt{d}$$

Experimental exploration [Srebro et al., 2006] I

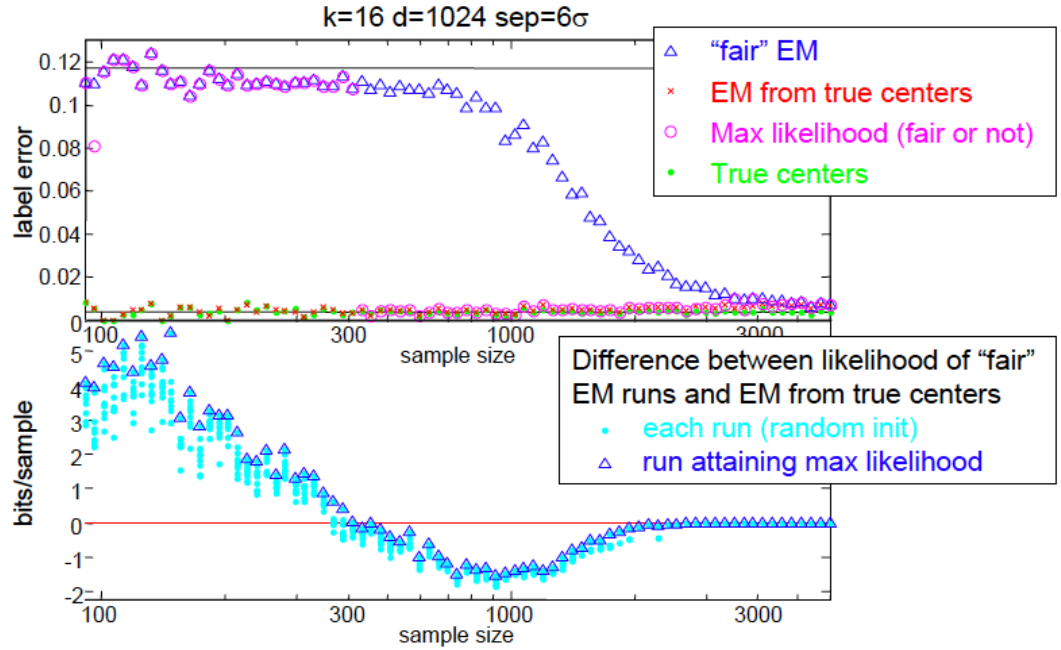
- High d
- True model: centers μ_k^* at corners of hypercube, $\Sigma_k^* = \sigma I_d$ spherical equal covariances, $\pi_k^* = 1/K$
- n, K, separation variable
- Algorithm: EM with Power initialization and projection on $(K-1)$-th principal subspace

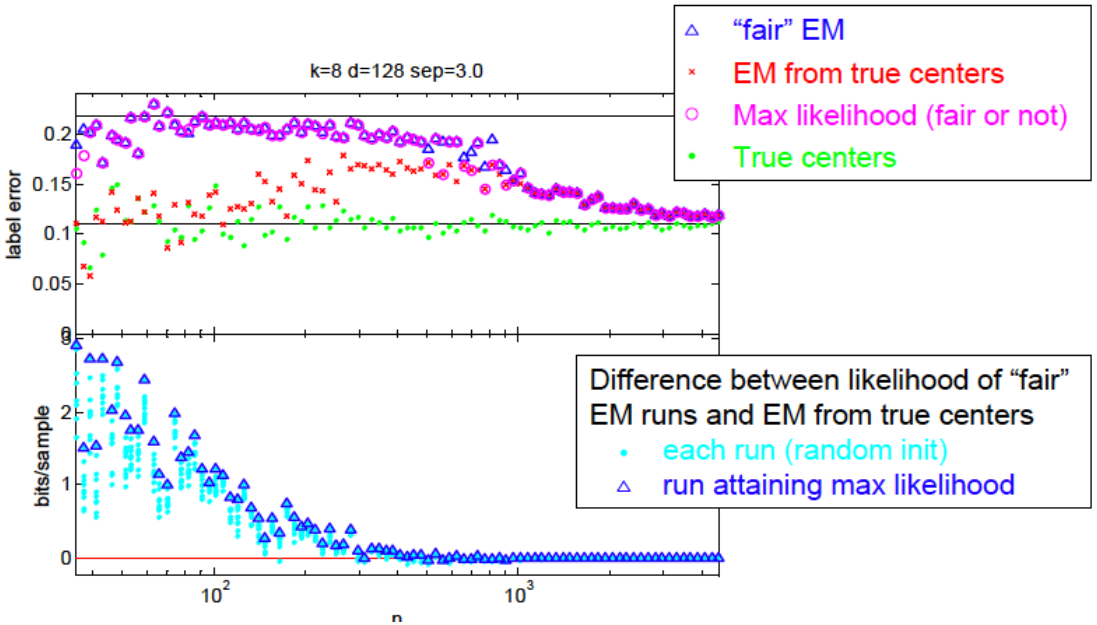
Experimental exploration [Srebro et al., 2006] II



figures from [Srebro et al., 2006]

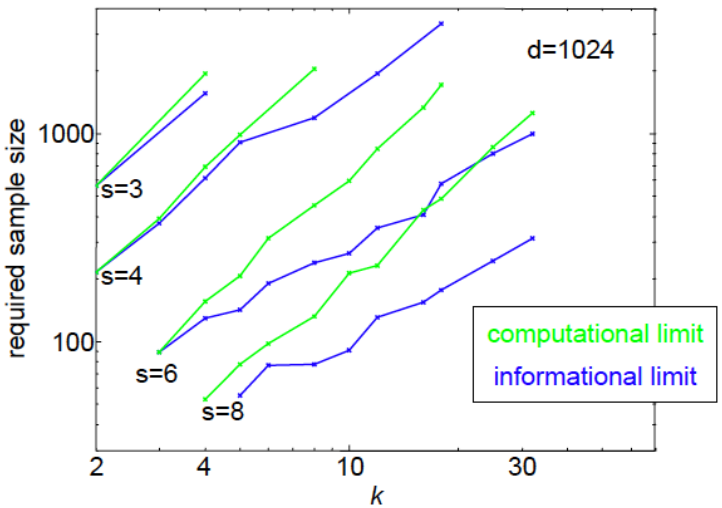
Experimental exploration [Srebro et al., 2006] III





Experimental exploration [Srebro et al., 2006] IV

- Practical limits vs theoretical limits



$n \propto k^{1.5} - k^{1.6}$ for all d, separation

figures from [Srebro et al., 2006]

Dasgupta 1999	$s > 0.5d^{1/2}$	$n = \Omega(k^{\log^2 1/\delta})$	Random projection, then mode finding
Dagupta Schulamn 2000	$s = \Omega(d^{1/4})$ (large d)	$n = poly(k)$	2 round EM with $\Theta(k \cdot \log k)$ centers
Arora Kannan 2001	$s = \Omega(d^{1/4} \log d)$		Distance based
Vempala Wang 2004	$s = \Omega(k^{1/4} \log dk)$	$n = \Omega(d^3k^2\log(dk/s\delta))$	Spectral projection, then distances

General mixture of Gaussians:

[Kannan Salmasian Vempala 2005] $s=\Omega(k^{5/2}\log(kd))$, $n=\Omega(k^2d\cdot\log^5(d))$

[Achliopts McSherry 2005] $s>4k+o(k)$, $n=\Omega(k^2d)$

Selecting K

- Run clustering algorithm for $K = K_{min} : K_{max}$
  - obtain $\Delta_{K_{min}}, \ldots \Delta_{K_{max}}$ or $\gamma_{K_{min}}, \ldots \gamma_{K_{max}}$
  - choose best Δ_K (or γ_K) from among them
- Typically increasing $K \Rightarrow$ cost $\mathcal{L}$ decreases
  - ($\mathcal{L}$ cannot be used to select K)
  - Need to "penalize" $\mathcal{L}$ with function of number parameters

Selecting K for mixture models

The BIC (Bayesian Information) Criterion

- let θ_K = parameters for γ_K
- let $\#\theta_K$=number independent parameters in θ_K
  - e.g for mixture of Gaussians with full Σ_k's in d dimensions

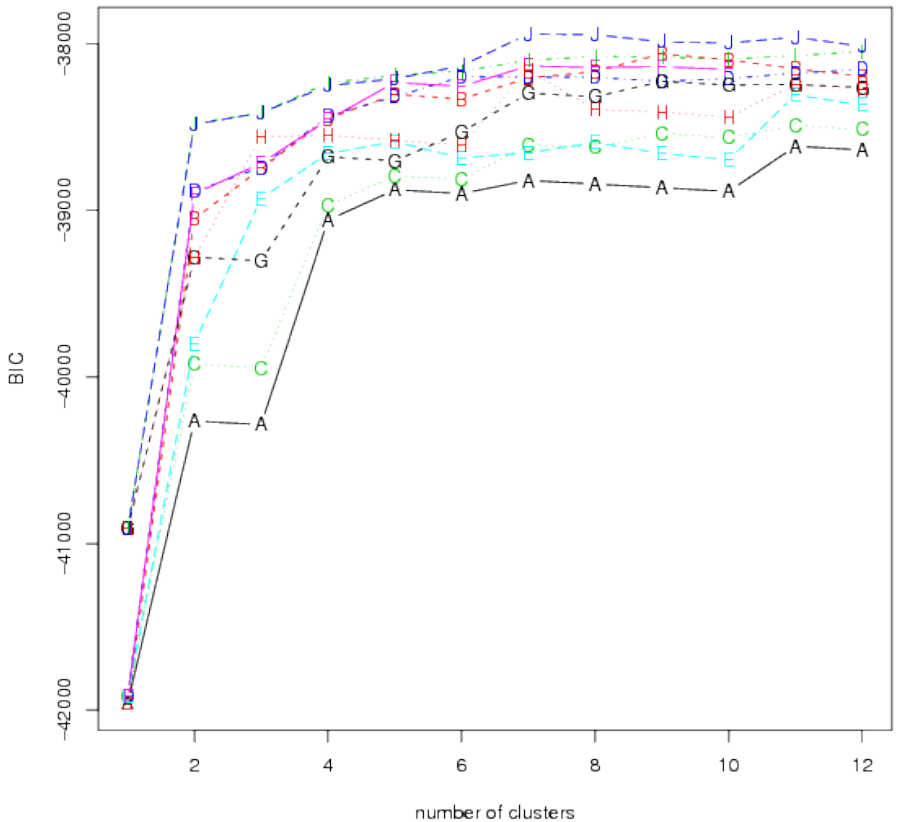
$$\#\theta_K = \underbrace{K-1}_{\pi_{1:K}} + \underbrace{Kd}_{\mu_{1:K}} + \underbrace{Kd(d-1)/2}_{\Sigma_{1:K}}$$

- define

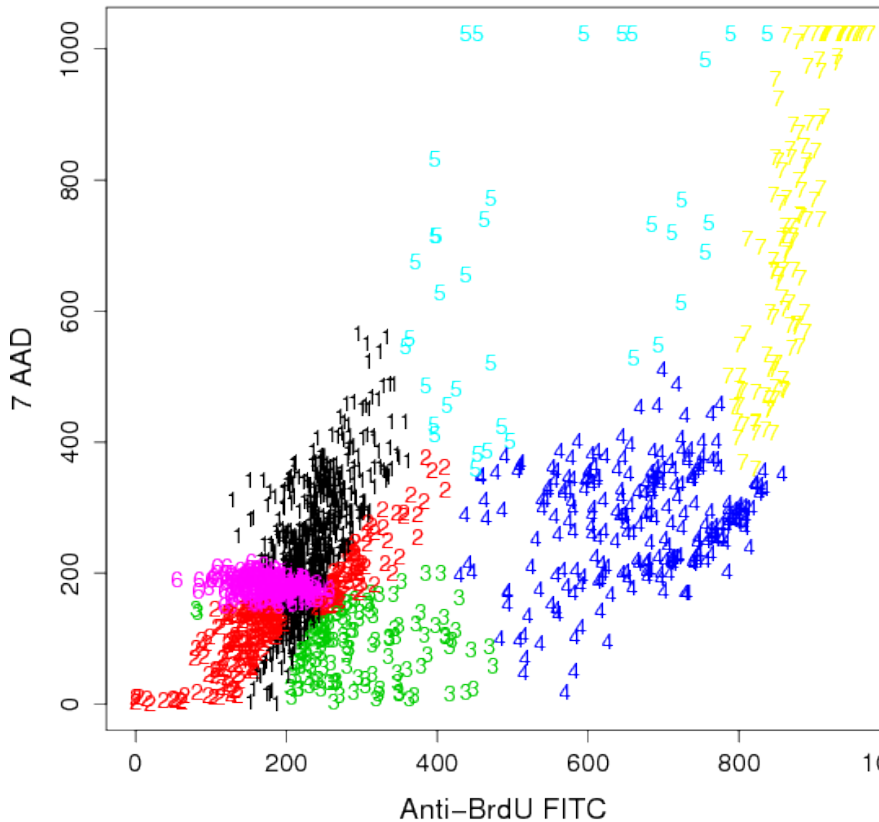
$$BIC(\theta_K) = l(\theta_K) - \frac{\#\theta_K}{2} \ln n$$

- **Select** K that maximizes $BIC(\theta_K)$
- selects true K for $n \to \infty$ and other technical conditions (e.g parameters in compact set)
- but theoretically not justified (and overpenalizing) for finite n

Number of Clusters vs. BIC EII (A), VII (B), EEI (C), VEI (D), EVI (E), VVI (F), EEE (G), EEV (H), VEV (I), VVV (J)

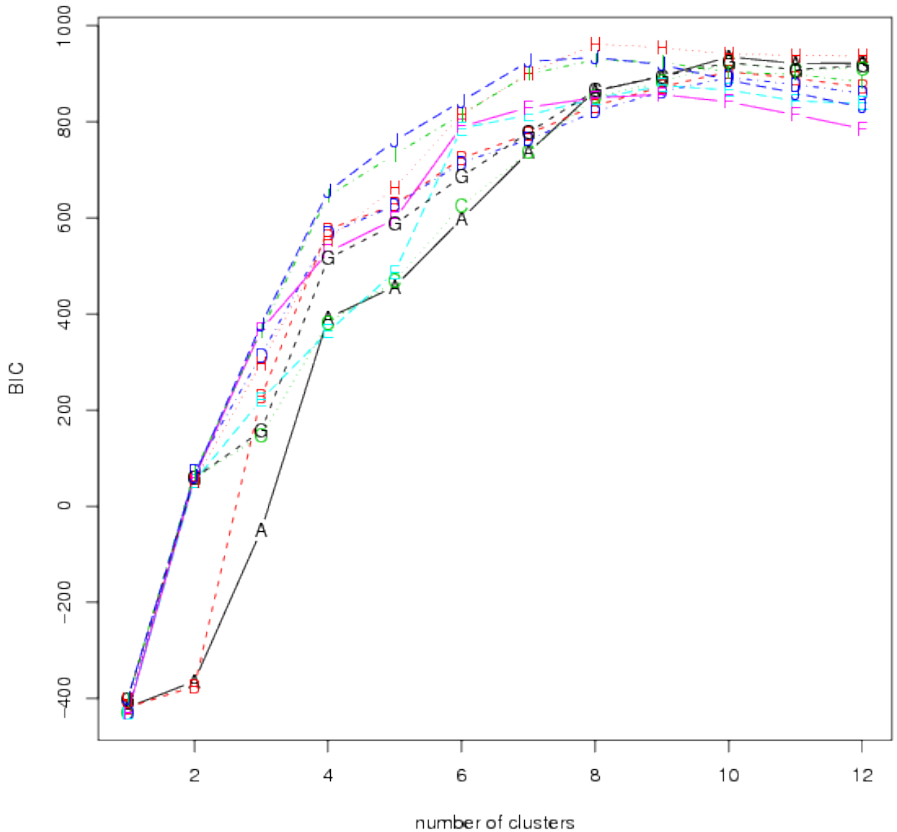


EEV, 8 Cluster Solution

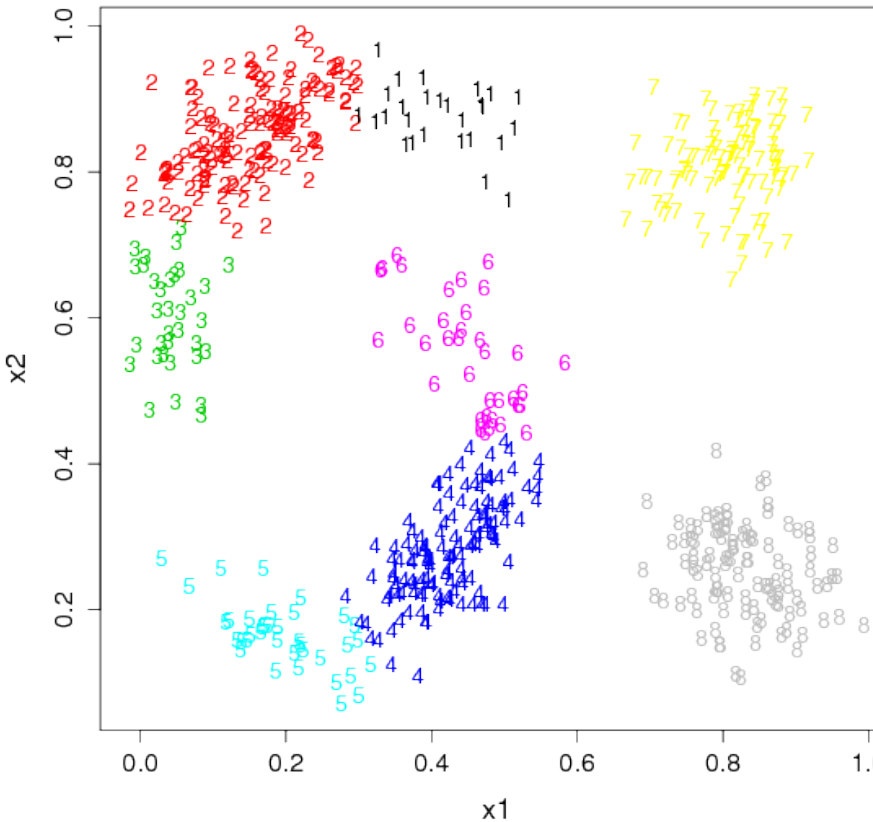


(from [Nugent and Meila, 2010])

Number of Clusters vs. BIC EII (A), VII (B), EEI (C), VEI (D), EVI (E), VVI (F), EEE (G), EEV (H), VEV (I), VVV (J)



EEV, 8 Cluster Solution



(from [Nugent and Meila, 2010])

Selecting K for hard clusterings

- based on statistical testing: the **gap** statistic (Tibshirani, Walther, Hastie, 2000)
- **X-means** [Pelleg and Moore, 2000] heuristic: splits/merges clusters based on statistical tests of Gaussianity
- Stability methods

The gap statistic

Idea

- for some cost $\mathcal{L}$ compare $\mathcal{L}(\Delta_K)$ with its expected value under a null distribution
  - choose null distribution to have no clusters
    - Gaussian (fit to data)
    - uniform with convex support
    - uniform over K_0 principal components of data
  - null value $= E_{P_0}[\mathcal{L}_{K,n}]$ the expected value of the cost of clustering n points from P_0 into K clusters
- the **gap**

$$g(K) = E_{P_0}[\mathcal{L}_{K,n}] - \mathcal{L}(\Delta_K) = \mathcal{L}_K^0 - \mathcal{L}(\Delta_K)$$

- choose K^* corresponding to the largest gap
- nice: it can also indicate that data has no clusters

Practicalities

- $\mathcal{L}_K^0 = E_{P_0}[\mathcal{L}_{K,n}]$ can rarely be computed in closed form (when P_0 very simple)
- otherwise, estimate $\mathcal{L}_K^0$ be Monte-Carlo sampling i.e generate B samples from P_0 and cluster them
- if sampling, variance s_K^2 of estimate $\hat{\mathcal{L}}_K^0$ must be considered s_K^2 is also estimated from the samples
- selection rule: $K^* =$ smallest K such that $g(K) \geq g(K+1) - s_{K+1}$
- favored $\mathcal{L}^V(\Delta) = \sum_k \frac{1}{|C_k|} \sum_{i \in C_k} ||x_i - \mu_k||^2 \approx$ sum of cluster variances

Stability methods for choosing K

- like bootstrap, or crossvalidation
- **Idea** (implemented by [Ben-Hur et al., 2002])

  for each K
  1. perturb data $\mathcal{D} \rightarrow \mathcal{D}'$
  2. cluster $\mathcal{D}' \rightarrow \Delta'_K$
  3. compare Δ_K, Δ'_K. Are they similar?
     If yes, we say Δ_K is **stable to perturbations**

Fundamental assumption If Δ_K is stable to perturbations then K is the correct number of clusters

- these methods are supported by experiments (not extensive)
- **not YET supported by theory** . . . see [von Luxburg, 2009] for a summary of the area

A stability based method for model-based clustering

- **The algorithm of [Lange et al., 2004]**
  1. divide data into 2 halves $\mathcal{D}_1$, $\mathcal{D}_2$ at random
  2. cluster (by EM) $\mathcal{D}_1 \rightarrow \Delta_1, \theta_1$
  3. cluster (by EM) $\mathcal{D}_2 \rightarrow \Delta_2, \theta_2$
  4. cluster $\mathcal{D}_1$ using $\theta_2 \rightarrow \Delta_1'$
  5. compare Δ_1, Δ_1'
  6. repeat B times and average the results

  - repeat for each K
  - select K where Δ_1, Δ_1' are closest on average (or most times)

Clustering with outliers

- What are outliers?
- let p = proportion of outliers (e.g 5%-10%)
- Remedies
  - mixture model: introduce a $K+1$-th cluster with large (fixed) Σ_{K+1}, bound Σ_k away from 0
  - K-means and EM
    - **robust** means and variances
      e.g eliminate smallest and largest $pn_k/2$ samples in mean computation (**trimmed mean**)
    - K-medians [Charikar and Guha, 1999]
    - replace Gaussian with a heavier-tailed distribution (e.g. Laplace)
  - single-linkage: do not count clusters with $< r$ points

Is K meaningful when outliers present?

- alternative: non-parametric clustering

Methods based on non-parametric density estimation

Idea The clusters are the isolated peaks in the (empirical) data density

- group points by the peak they are under
- some outliers possible
- $K = 1$ possible(no clusters)
- shape and number of clusters K determined by algorithm
- **structural parameters**
  - **smoothness** of the **density estimate**
  - what is a peak

Algorithms

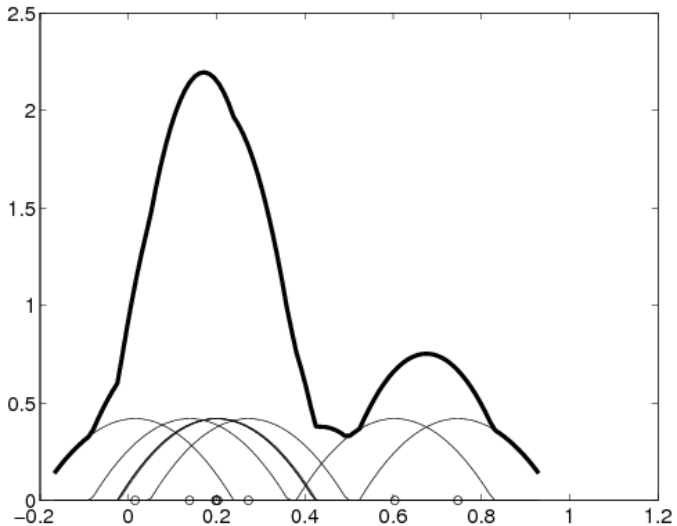
- peak finding algorithms Mean-shift algorithms
- level sets based algorithms
  - Nugent-Stuetzle, Support Vector clustering
- Information Bottleneck [Tishby and Slonim, 2000]

Kernel density estimation

Input
- data $\mathcal{D} \subseteq \mathbb{R}^d$
- **Kernel** function $K(z)$
- parameter **kernel width** h (is a **smoothness parameter**)

Output $f(x)$ a **probability density** over $\mathbb{R}^d$

$$f(x) = \frac{1}{Nh^d} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$



- f is sum of Gaussians centered on each x_i
- f is smoother (less variation) if h larger
- **caveat:** dimension d can't be too large

The kernel function

- Example $K(z) = \frac{1}{(2\pi)^{d/2}} e^{-||z||^2/2}$, $z \in \mathbb{R}^d$ is the Gaussian kernel
- In general
  - $K()$ should represent a density on $\mathbb{R}^d$, i.e $K(z) \geq 0$ for all z and $\int K(z)dz = 1$
  - $K()$ symmetric around 0, decreasing with $||z||$
- In our case, K must be differentiable

Mean shift algorithms

Idea find points with $\nabla f(x) = 0$

Assume $K(z) = e^{-||z||^2/2}/\sqrt{2\pi}$ Gaussian kernel

$$\nabla f(x) = -\frac{1}{Nh^d} \sum_{i=1}^{n} K(\frac{x - x_i}{h})(\mathbf{x} - x_i)/h$$

Local max of f is solution of implicit equation

$$x = \underbrace{\frac{\sum_{i=1}^{n} x_i K(\frac{x-x_i}{h})}{\sum_{i=1}^{n} K(\frac{x-x_i}{h})}}_{\text{the } \mathbf{mean\ shift} m(x)}$$

Algorithm Simple Mean Shift

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, kernel $K(z)$, h

1. for $i = 1 : n$
   - 1.1 $x \leftarrow x_i$
   - 1.2 iterate $x \leftarrow m(x)$ until convergence to m_i
2. group points with same m_i in a cluster

Remarks

- mean shift iteration guaranteed to converge to a max of f
- computationally expensive
- a faster variant...

Algorithm Mean Shift (Comaniciu-Meer)

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, kernel $K(z)$, h

1. select q points $\{x_j\}_{j=1:q} = \mathcal{D}_q \subseteq \mathcal{D}$ that cover the data well
2. for $j \in \mathcal{D}_q$
   - 2.1 $x \leftarrow x_j$
   - 2.2 iterate $x \leftarrow m(x)$ until convergence to m_j
3. group points in $\mathcal{D}_q$ with same m_j in a cluster
4. assign points in $\mathcal{D} \setminus \mathcal{D}_q$ to the clusters by the **nearest-neighbor** method

$$k(i) = k(\operatorname*{argmin}_{j \in \mathcal{D}_q} ||x_i - x_j||)$$

Gaussian blurring mean shift

Idea

- like **Simple Mean Shift** but points are shifted to new locations
- the density estimate f changes
- becomes concentrated around peaks very fast

Algorithm Gaussian Blurrring Mean Shift (GBMS)

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, Gaussian kernel $K(z)$, h

1. Iterate until STOP
   - 1.1 for $i = 1 : n$ compute $m(x_i)$
   - 1.2 for $i = 1 : n$, $x_i \leftarrow m(x_i)$

Remarks

- all x_i converge to a single point

  $\Rightarrow$ need to stop before convergence

Empirical stopping criterion [Carreira-Perpinan, 2007]

- define $e_i^t = ||x_i^t - x_i^{t-1}||$ the change in x_i at t
- define $H(e^t)$ the **entropy** of the **histogram** of $\{e_i^t\}$
- STOP when $\sum_{i=1}^n e_i^t/n <$`tol` OR $|H(e^t) - H(e^{t-1})| <$`tol`'

Convergence rate If true f Gaussian, convergence is **cubic**

$$||x_i^t - x^*|| \leq C||x_i^{t-1} - x^*||^3$$

very fast!!

Algorithm Nugent-Stuetzle

Input Data $\mathcal{D} = \{x_i\}_{i=1:n}$, kernel $K(z)$

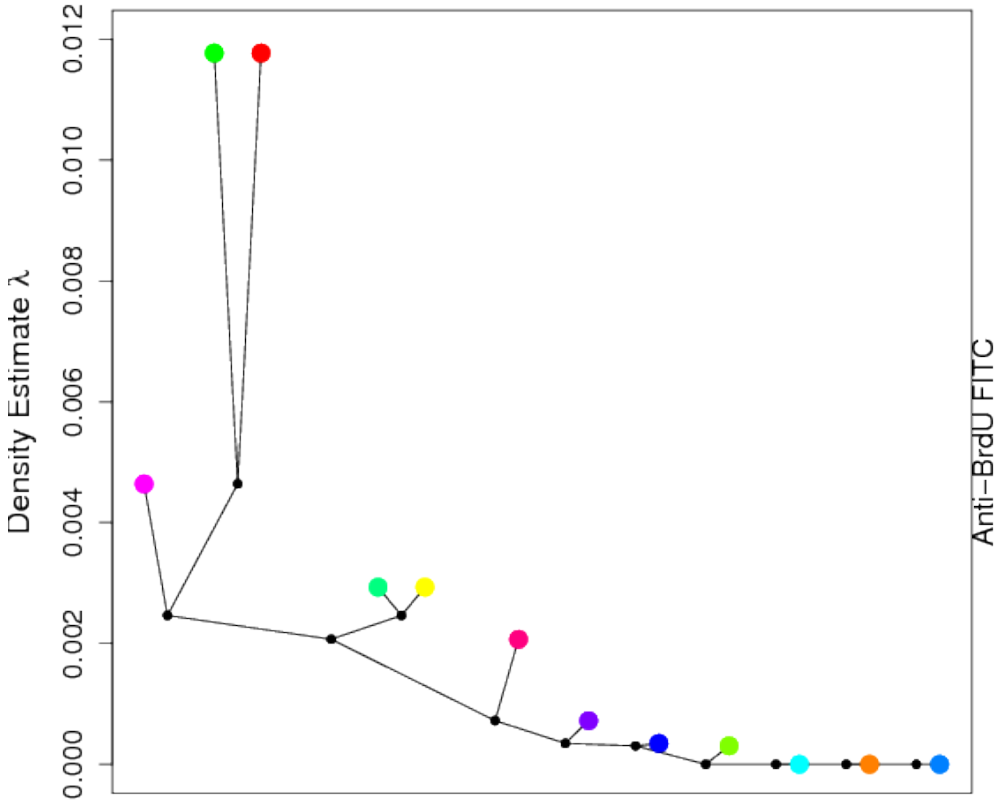
1. Compute KDE $f(x)$ for chosen h
2. for levels $0 < l_1 < l_2 < \ldots < l_r < \ldots < l_R \geq \sup_x f(x)$
   - 2.1 find **level set** $L_r = \{x \mid f(x) \geq l_r\}$ of f
   - 2.2 if L_r **disconnected** then each connected component is a cluster $\rightarrow (C_{r,1}, C_{r,2}, \ldots C_{r,K_r})$

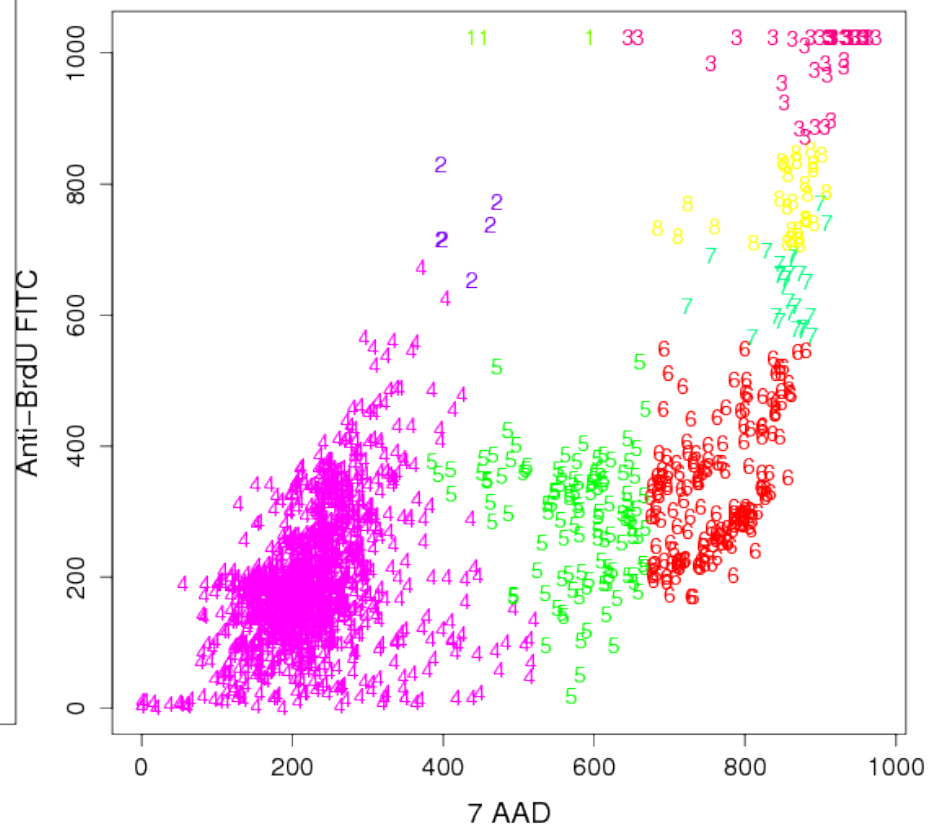
Output clusters $\{(C_{r,1}, C_{r,2}, \ldots C_{r,K_r})\}_{r=1:R}$

Remarks

- every cluster $C_{r,k} \subseteq$ some cluster $C_{r-1,k'}$
- therefore output is hierarchical clustering
- some levels can be pruned (if no change, i.e. $K_r = K_{r-1}$)
- algorithm can be made recursive, i.e. efficient
- finding level sets of f tractable only for $d = 1, 2$
- for larger d, $L_r = \{x_i \in \mathcal{D} \mid f(x_i) \geq l_r\}$
- to find connected components
  - for $i \neq j \in L_r$
    if $f(tx_i + (1-t)x_j) \geq l_r$ for $t \in [0, 1]$
    then $k(i) = k(j)$
- confidence intervals possile by resampling

Cluster tree with 13 leaves (8 clusters, 5 artifacts)





(from [Nugent and Meila, 2010])

Chaudhuri-Dasgupta Algorithm

- Uses k-**nearest neighbor** graphs (filtration)
- Parameters k (nearest neigbhors) and $\alpha \in [1, 2]$
- for $r \geq 0$, $G_r = (V_r, E_r)$ with
  - $x_i \in V_r$ iff **distance to k-nn of x_i** $\leq r$
  - $(x_i, x_j) \in E_r$ iff $||x_i - x_j|| \leq \alpha r$

Consistency Theorem For any ϵ (separation parameter) and δ (confidence), $\alpha \in [\sqrt{2}, 2]$ (graph density), if $k = C \log^2(1/\delta) \frac{d \log n}{\epsilon^2}$
for any two clusters C, C' in cluster tree, there exists a level r so that $C \cap \mathcal{D}, C' \cap \mathcal{D}$ are clusters at level r

Chaudhuri-Dasgupta Algorithm

Consistency Theorem For any ϵ (separation parameter) and δ (confidence), $\alpha \in [\sqrt{2}, 2]$ (graph density), if $k = C \log^2(1/\delta) \frac{d \log n}{\epsilon^2}$
for any two clusters C, C' in cluster tree, there exists a level r so that $C \cap \mathcal{D}, C' \cap \mathcal{D}$ are clusters at level r

- r depends on $\lambda =$"bridge" between C, C' (and $\sigma > 0$ "tube" width)

$$r^d \omega_d \lambda = \frac{k}{\mathbf{n}} + \ldots \textit{confidence term}$$

- it follows that the needed sample size n at level λ

$$n = \mathcal{O}\left(\frac{d}{\lambda \epsilon^2 (\sigma/2)^d \omega_d} \log \frac{d}{\lambda \epsilon^2 (\sigma/2)^d \omega_d} \right)$$

- this **sample complexity** n is almost tight
- for $\alpha < \sqrt{2}$ sample complexity is exponential in d

- New results **[Kent, B. P., Rinaldo, A. and Verstynen, T. 2013]**

- Remark: algorithm(s) can be applied in any metric space

Support Vector (SV) clustering

Idea same as for Nugent-Stuetzle, but use **kernelized density estimator** instead of KDE

Algorithm SV

Input data $\mathcal{D}$, parameters q kernel width, $p \in (0,1)$ proportion of outliers

1. construct a 1-class SVM with parameters q, $C = 1/np$
   this is equivalent to enclosing the data in a sphere in feature space
   for any x its distance from center of sphere is

$$R^2(x) = K(x,x) - 2\sum_j \alpha_j K(x,x_j) + \sum_{i,j} K(x_i,x_j)$$

   for x_i support vector, $R(x_i) = R$ (same for all)
2. for all pairs $i,j = 1:n$
   - i,j in same cluster if segment $[i,j]$ is within sphere with radius R in feature space
   - practically, test if $R(tx_i + (1-t)x_j) < R$ for t on a grid over [0,1]

Remarks

- the **kernel** used by SV is $K(x, x') = e^{-q||x-x'||^2}$
- q controls boundary smoothness
- SV's lie on cluster boundaries, "margin error" points lie outside clusters (are outliers)
- SV theory $\frac{\text{margin errors}}{n} \rightarrow \frac{1}{nC} = p$ for large n
- hence p controls the proportion of outliers
- p, q together control K
  p larger, q smaller $\Rightarrow$ K smaller

The Dirichlet distribution

- $Z \in \{1 : r\}$ a discrete random variable, let $\theta_j = P_z(j)$, $j = 1, \ldots r$.
- **Multinomial distribution** Probability of i.i.d. sample of size N from P_z

$$P(z^{1,\ldots N}) = \prod_{j=1}^{r} \theta_j^{N_j}$$

  where $N_j = \#$the value j is observed, $j = 1, \ldots r$
- $N_{1:r}$ are the **sufficient statistics** of the data.
- The **Dirichlet distribution** is defined over domain of $\theta_{1,\ldots r}$, with real parameters $N'_{1,\ldots r} > 0$ by

$$D(\theta_{1,\ldots r}; N'_{1,\ldots r}) = \frac{\Gamma(\sum_j N'_j)}{\prod_j \Gamma(N'_j)} \prod_j \theta_j^{N'_j - 1}$$

  where $\Gamma(p) = \int_0^\infty t^{p-1} e^{-t} dt$.

Dirichlet process mixtures

- Model-based
- generalization of mixture models to
  - infinite K
  - Bayesian framework
- denote θ_k = parameters for component f_k
- assume $f_k(x) \equiv f(x, \theta_k) \in \{f(x, \theta)\}$
- assume prior distributions for parameters $g_0(\theta)$
- prior with hyperparameter $\alpha > 0$ on the number of clusters
- very flexible model

A sampling model for the data

- **Example: Gaussian mixtures**, $d = 1$, $\sigma_k = \sigma$ fixed
- $\theta = \mu$
- prior for μ is $Normal(0, \sigma_0^2 I_d)$
- Sampling process
  - for $i = 1 : n$ sample $x_i, k(i)$ as follows

    denote $\{1 : K\}$ the clusters after step $i - 1$
    define n_k the size of cluster k after step $i - 1$

    1. $$k(i) = \begin{cases} k & \text{w.p } \frac{n_k}{i-1+\alpha}, \ k = 1 : K \\ K+1 & \text{w.p } \frac{\alpha}{i-1+\alpha} \end{cases} \tag{15}$$
    2. if $k(i) = K + 1$ sample $\mu_i \equiv \mu_{K+1}$ from $Normal(0, \sigma_0^2)$
    3. sample x_i from $Normal(\mu_{k(i)}, \sigma^2)$
- can be shown that the distribution of $x_{1:n}$ is **interchangeable** (does not depend on data permutation)

The hyperparameters

- σ_0 controls spread of centers
  - should be large
- α controls number of cluster centers
  - α large $\Rightarrow$ many clusters
- cluster sizes non-uniform (larger clusters attract more new points)
- many single point clusters possible

General Dirichlet mixture model

- cluster densities $\{f(x, \theta)\}$
- parameters θ sampled from prior $g_0(\theta, \beta)$
- cluster membership $k(i)$ sampled as in (15)
- x_i sampled from $f(x, \theta_{k(i)})$
- **Model Hyperparameters** α, β

Clustering with Dirichlet mixtures

The clustering problem

- $\alpha, g_0, \beta, \{f\}$ given
- $\mathcal{D}$ given
- wanted $\theta_{1:n}$ (not all distinct!)
- note:
  - $\theta_{1:n}$ determines a hard clustering Δ

Estimating $\theta_{1:n}$ cannot be solved in closed form
Usually solved by MCMC (Markov Chain Monte Carlo) sampling

Clustering with Dirichlet mixtures via MCMC

MCMC estimation for Dirichlet mixture

Input $\alpha, g_0, \beta, \{f\}, \mathcal{D}$

State cluster assignments $k(i)$, $i = 1:n$,
parameters θ_k for all distinct k

Iterate

1. for $i = 1:n$ (reassign data to clusters)
   - 1.1 resample $k(i)$ by

$$k(i) = \begin{cases} \text{existing } k & \text{w.p } \propto \frac{n_k - 1}{n-1+\alpha} f(x_i, \theta_k) \\ \text{new cluster} & \text{w.p } \frac{\alpha}{n-1+\alpha} \int f(x_i, \theta) g_0(\theta) d\theta \end{cases} \tag{16}$$

   - 1.2 if $k(i)$ is new label, sample a new $\theta_{k(i)} \propto g_0 f(x_i, \theta)$
2. for $k \in \{k(1:n)\}$ (resample cluster parameters)
   - 2.1 sample θ_k from posterior $g_k(\theta) \propto g_0(\theta, \beta) \prod_{i \in C_k} f(x_i, \theta)$

   g_k can be computed in closed form if g_0 is conjugate prior

Output a state with high posterior

Summary: Parametric vs. non-parametric

Parametric clustering

- Optimizes a cost $\mathcal{L}$
- Most costs are NP-hard to optimize
- Assumes more detailed knowledge of cluster shapes
- Assumes K known (But there are wrapper methods to select K)
- gets harder with larger K
- Older, more used and studied

Non-parametric clustering

- density-based methods have no cost function

  Dirichlet clustering samplers posterior of $k(1:n), \{\theta_k\}$ given $\mathcal{D}$
- do not depend critically on initialization
- K and outliers selected automatically, naturally
- require hyperparameters (= smoothness parameters)

Note that **Dirichlet mixture** is inbetween parametric and non-parametric

When to use

- Parametric
  - shape of clusters known
  - K not too large or known
  - clusters of comparable sizes
- Non-parametric (density based)
  - shape of clusters arbitrary
  - K large or many outliers
  - clusters sizes in large range (a few large clusters and many small ones)
  - dimension d small (except for **SV**)
  - lots of data
- Dirichlet mixtures
  - shape of clusters known
  - clusters sizes in large range

Similarity based clustering

- **Paradigm:** the features we observe are measures of **similarity/dissimilarity** between pairs of data points, e.g

	points	features
Image segmentation	pixels	distance in color space or location, separated by a contour, belong to same texture
Social network	people	friends, coworkers, phone calls, emails
Text analysis	words	appear in same context

- The features are summarized by a single **similarity measure** S_{ij}
  - e.g $S_{ij} = e^{\sum_k \alpha_k \text{feature}_k(i,j)}$ for all points i, j
  - symmetric $S_{ij} = S_{ji}$
  - non-negative $S_{ij} \geq 0$
- We want to put points that are similar to each other in the same cluster, dissimilar points in different clusters
- Problem is often cast as a **graph cut** problem
  - points = graph nodes, similarity S_{ij} = weight of edge ij
  -

Paradigms for grouping

- Graph cuts

  remove some edges $\Longrightarrow$ disconnected graph

  the groups are the connected components

- By similar behavior

  nodes i, j in the same group iff i, j have the same pattern of connections w.r.t other nodes

- By Embedding

- map nodes $V = \{1, 2, \ldots, n\} \longrightarrow \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^d$ then use standard classification and clustering methods

Definitions

- $V = \{1, 2, \ldots, n\}$
- node **degree** or **volume**

$$D_i = \sum_{j \in V} S_{ij}$$

- **volume** of cluster $C \subseteq V$

$$D_C = \sum_{i \in C} D_i$$

- **cut** between subsets $C, C' \subseteq V$

$$\sum_{i \in C} \sum_{j \in C'} S_{ij}$$

- **Multiway Normalized Cut** of a partition $\Delta = \{C_{1:K}\}$ of V

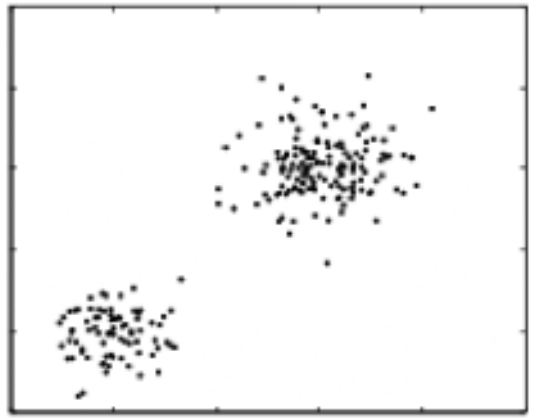
$$MNCut(\Delta) = \sum_{k=1}^{K} \sum_{k' \neq k} \frac{Cut(C_k, C_{k'})}{D_{C_k}}$$

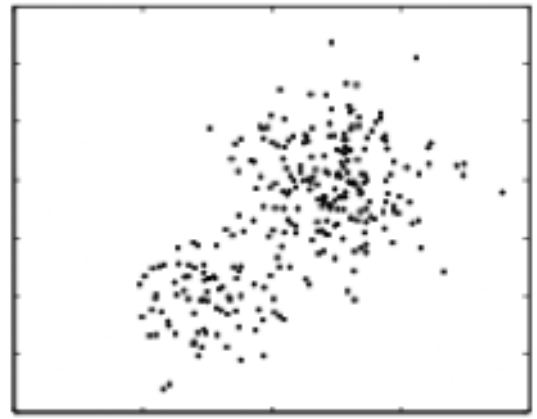
in particular, for $K = 2$,

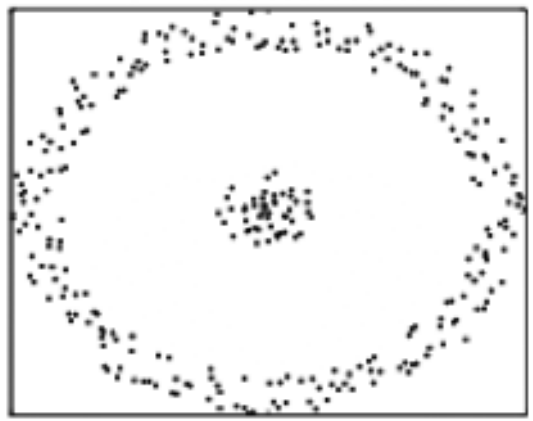
$$MNCut(C, C') = Cut(C, C') \left(\frac{1}{D_C} + \frac{1}{D_{C'}} \right)$$

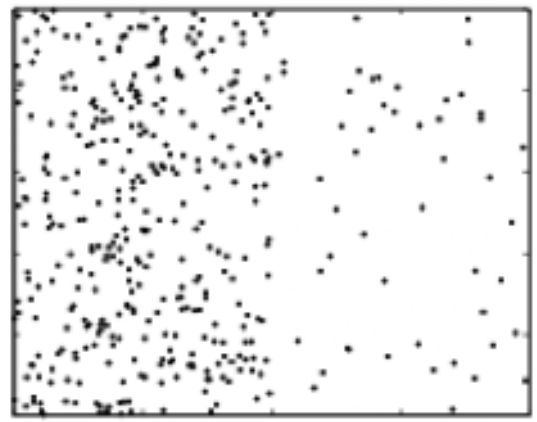
Motivation for MNCut

$S_{ij} \propto 1/\text{dist}(i,j)$

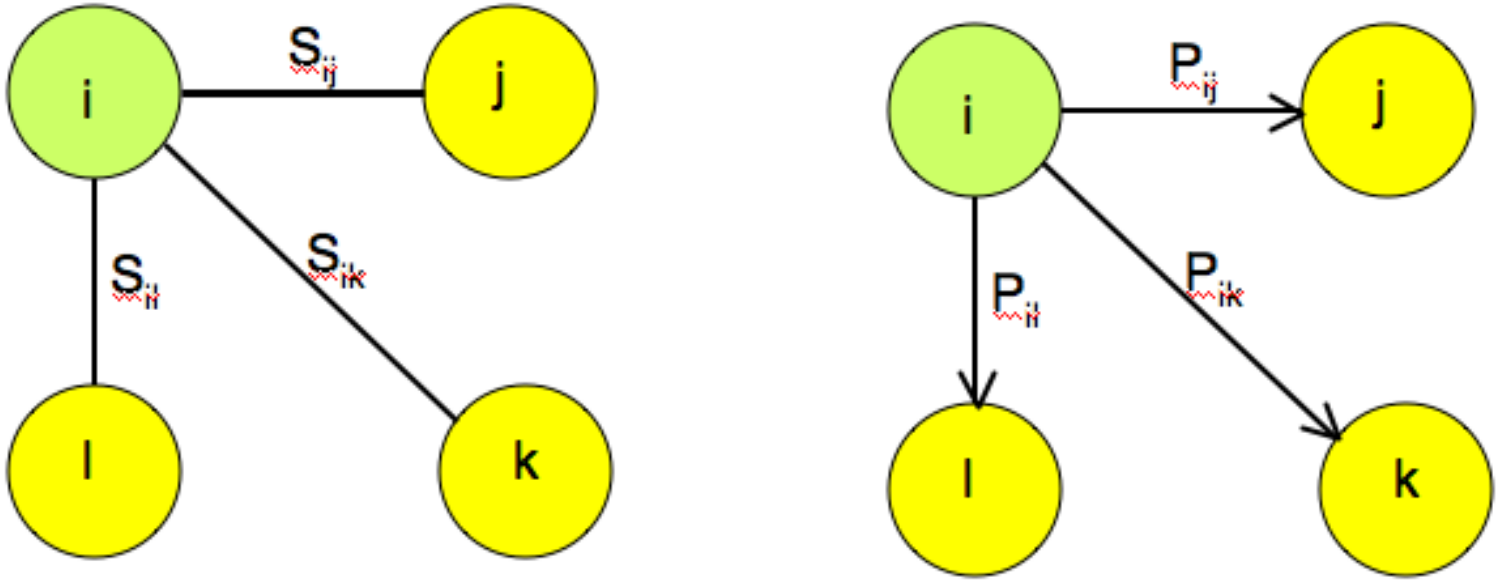








A random walks view



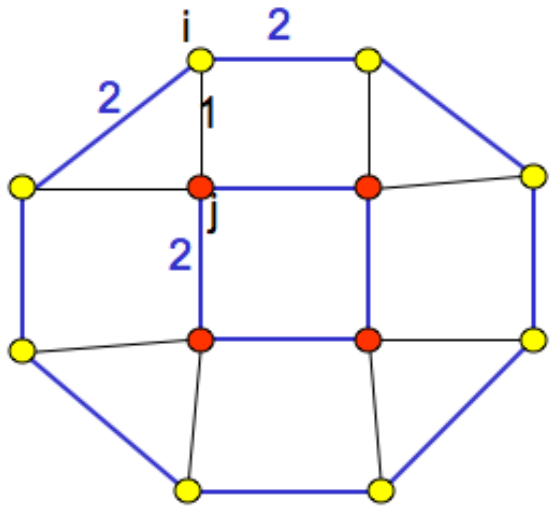
- Define

$$P_{ij} = \frac{S_{ij}}{D_i} \quad \text{for all } i, j \in V$$

- in matrix notation $P = D^{-1}S$ where $P = [P_{ij}]$, $D = \text{diag}(D_1, \ldots D_n)$
- P defines a **random walk** over the graph nodes V

Grouping from the random walks point of view

- Idea: group nodes together if they transition in the same way to other clusters



$$P_{i,red} = Pr[\, i \rightarrow red \,|\, i] = \sum_{j \in red} P_{ij}$$

i	$P_{i,red}$	$P_{i,yellow}$
●	1/5	4/5
●	1/5	4/5
●	1/5	4/5
●	1/5	4/5
●	1/5	4/5
●	1/5	4/5
●	1/5	4/5
●	1/5	4/5
●	2/3	1/3
●	2/3	1/3
●	2/3	1/3
●	2/3	1/3

... is the same as grouping by embedding

- **embedding** of V = mapping from V into $\mathbb{R}^d$
- Wanted: similar points embedded near each other

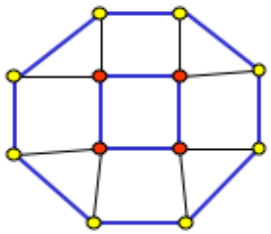
  ideally, points in the same cluster mapped to the same point in $\mathbb{R}^d$

Another look at $P_{i,c}$

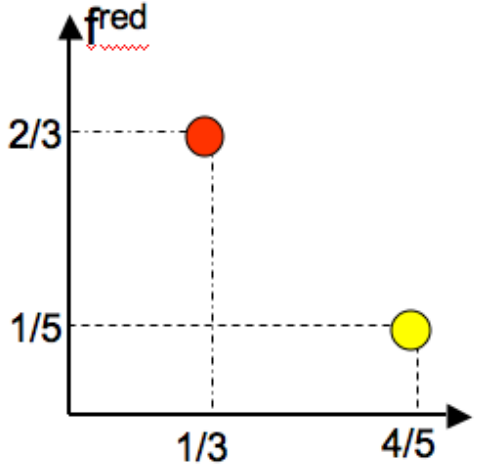
$P_{\cdot,red} \equiv f^{red}$

$P_{\cdot,yel} \equiv f^{yel}$

a **piecewise constant** function

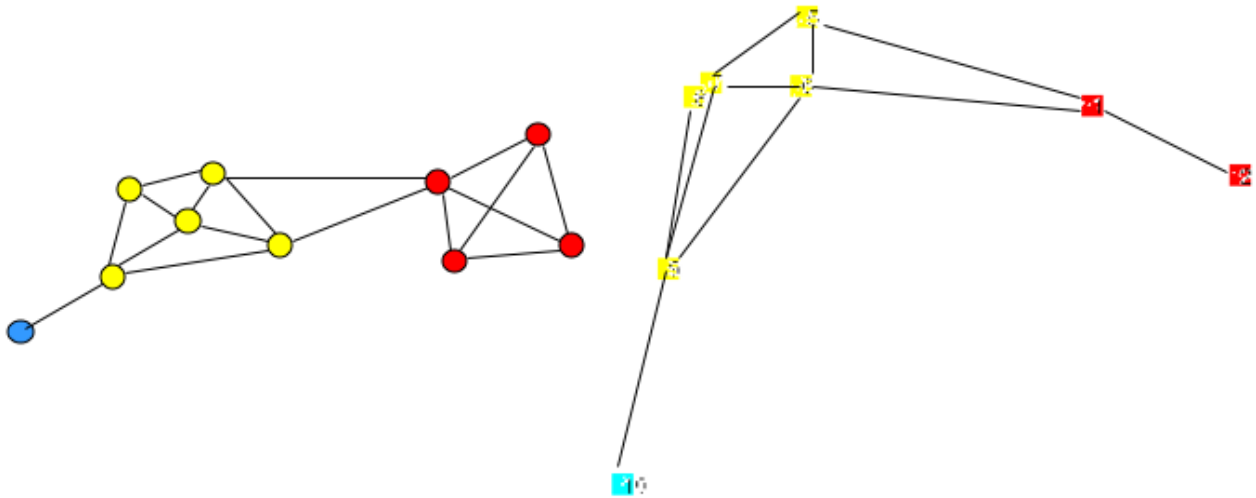


i	$P_{i,red}$	$P_{i,yellow}$
	1/5	4/5
	1/5	4/5
	1/5	4/5
	1/5	4/5
	1/5	4/5
	1/5	4/5
	1/5	4/5
	1/5	4/5
	2/3	1/3
	2/3	1/3
	2/3	1/3
	2/3	1/3



Some questions

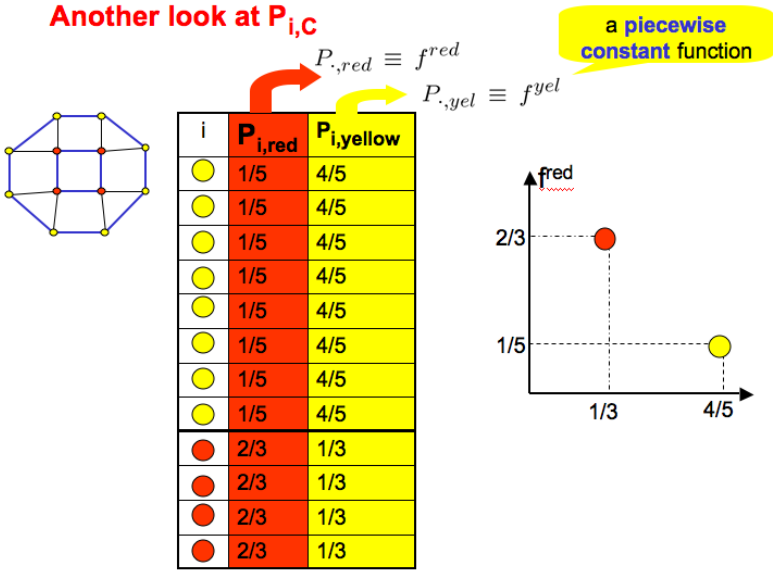
- Not all graphs embed perfectly



- How many dimensions do we need?
- Nice, but we need to know the clusters in advance...

Lumpability

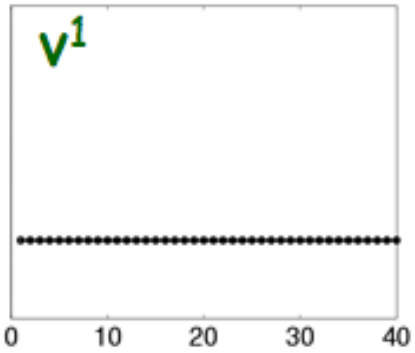
- A vector v is **piecewise constant** w.r.t a clustering Δ iff $v_i = v_j$ whenever i, j in same $C \in \Delta$

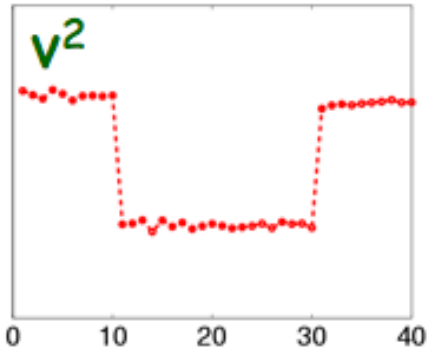


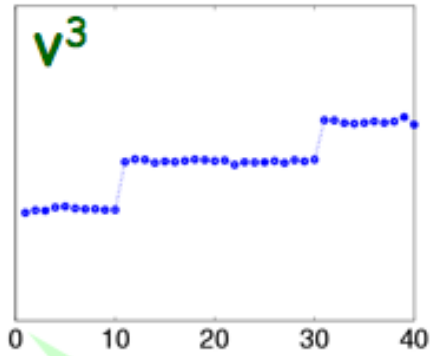
- **Theorem [Lumpability]**[Meila&Shi 2001] Let S be a similarity matrix and Δ a clustering with K clusters. Then P has K **piecewise constant** eigenvectors w.r.t Δ **iff**

$$\sum_{j \in C'} P_{ij} = R_{CC'} \text{ whenver } i \in C, \text{ for all } C, C' \in \Delta$$

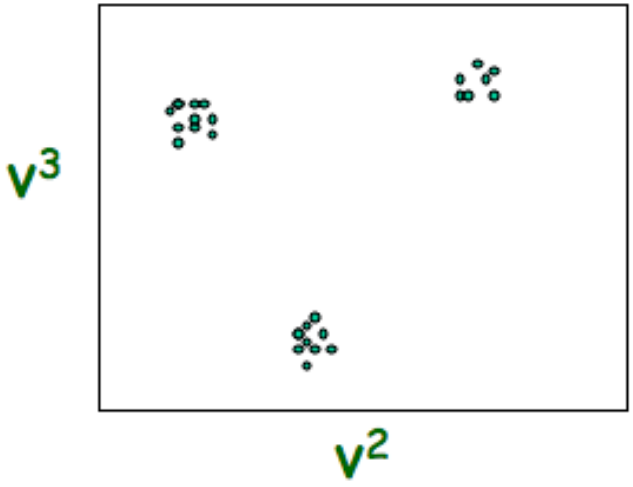
The spectral mapping





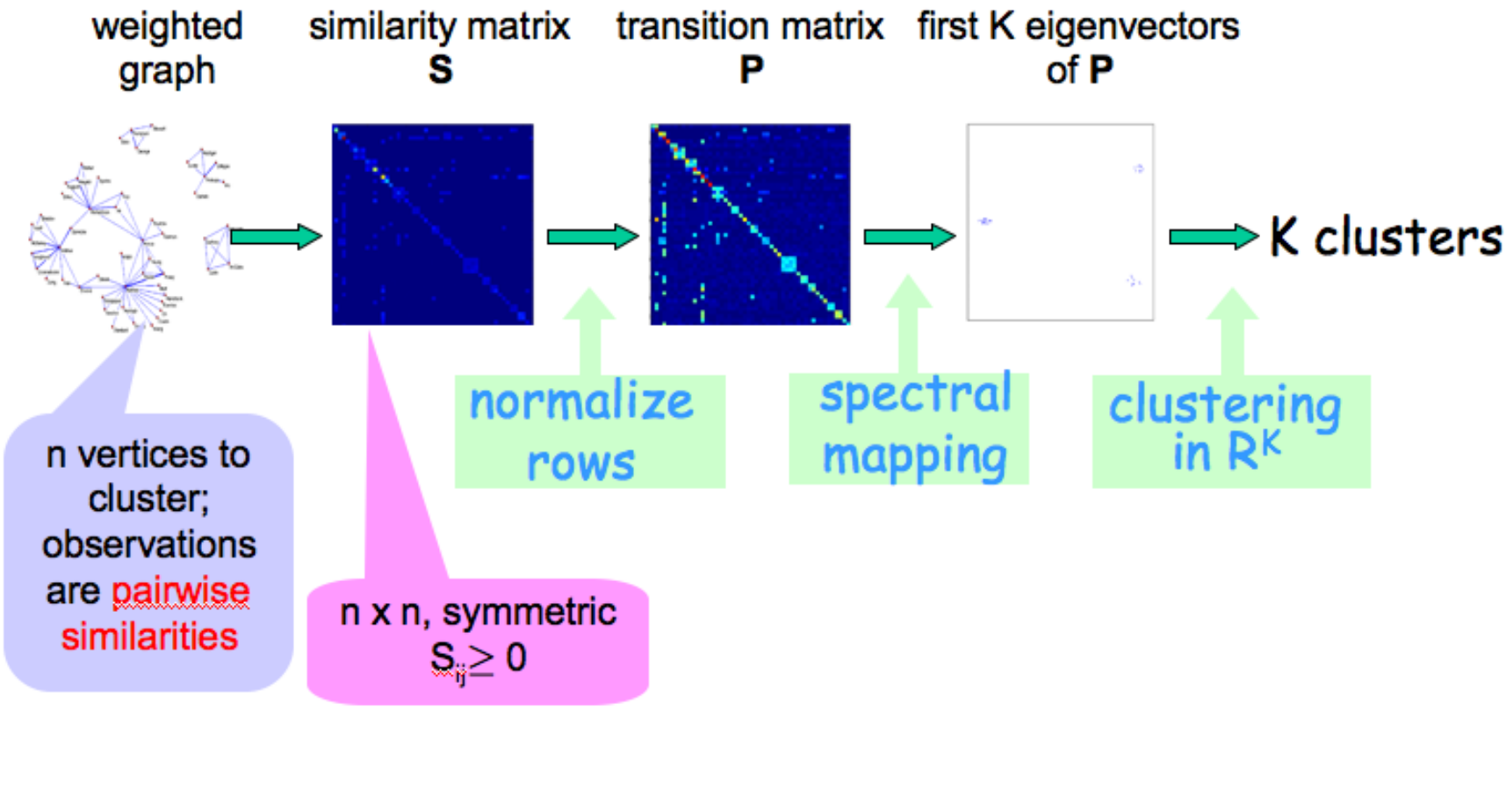


The **spectral mapping**: Data as elements of v^2, v^3



These eigenvectors are called **piecewise constant (PC)**

Spectral clustering in a nutshell



Spectral clustering

An algorithm based on [Meilă and Shi, 2001b] and [Ng et al., 2002].

Spectral Clustering Algorithm

Input Similarity matrix S, number of clusters K

1. Transform S: Set $D_i = \sum_{j=1}^{n} S_{ij}$, $j = 1 : n$ the **node degrees**.
   Form the **transition matrix** $P = [P_{ij}]_{ij=1}^{n}$ with

$$P_{ij} \leftarrow S_{ij}/D_i, \text{ for } i, j = 1 : n$$

2. Compute the largest K eigenvalues $\lambda_1 = 1 \geq \lambda_2 \geq \ldots \geq \lambda_K$ and eigenvectors $\mathbf{v}_1, \ldots \mathbf{v}_K$ of P.
3. Embed the data in principal subspace Let $V = [\mathbf{v}_2\ \mathbf{v}_3\ \ldots\ \mathbf{v}_K] \in \mathbb{R}^{n \times K}$, $\mathbf{x}_i \leftarrow$ i-th row of V.
4. (orthogonal initialization) Find K initial centers by
   4.1 take μ_1 randomly from $\mathbf{x}_1, \ldots \mathbf{x}_n$
   4.2 for $k = 2, \ldots K$ set $\mu_k = \operatorname{argmin}_{\mathbf{x}_i} \max_{k' < k} \mu_{k'}^T \mathbf{x}_i$.
5. Run the K-means algorithm on the "data" $\mathbf{x}_{1:n}$ starting from the centers $\mu_{1:K}$.

Properties of spectral clustering

- Arbitrary cluster shapes (main advantage)
- Elegant mathematically
- Practical up to medium sized problems
  - Running time (by Lanczos algorithm) $\mathcal{O}(nk)$/iteration.
- Works well when K known, not too large

  estimating K [Azran and Ghahramani, 2006]
- Depend heavily on the similarity function (main problem)

  learning the similarities
  [Meilă and Shi, 2001a],[Bach and Jordan, 2006],[Meilă et al., 2005],[Shortreed and Meilă,
- Outliers become separate clusters (user must adjust K accordingly!)
- Very popular, many variants which aim to improve on the above

  Diffusion maps [Nadler et al., 2006]: normalize the eigenvectors $\lambda_k^t v^k$
- Practical fix, when K large: only compute a fixed number of eigenvectors $d < K$.
  This avoids the effects of noise in lower ranked eigenvectors

Affinity propagation

- **Idea** Each item $i \in \mathcal{D}$ finds an **exemplar** item $k \in \mathcal{D}$ to "represent" it
- Affinity Propagation is to spectral clustering what Mean Shift is to K-means
- number of exemplars not fixed in advance
- quantities of interest
  - similarities s_{ij}, $i \neq j$ (given)
  - **availability** a_{ik} of k for i = how much support there is from other items for k to be an exemplar
  - **responsibility** r_{ik} that measures how fit is k to represent i, as compared to other possible candidates k'.
  - diagonal elements s_{ii} represent **self-similarities**
    - larger s_{ii} $\Rightarrow$ more likely i will become an exemplar $\Rightarrow$ more clusters

Affinity Propagation

Affinity Propagation Algorithm [Frey and Dueck, 2007]

Input Similarity matrix $S = [s_{ik}]_{ik=1}^{n}$, parameter $\lambda = 0.5$

Iterate the following steps until convergence

1. $a_{ik} \leftarrow 0$ for $i, k = 1 : n$
2. for all i
   - 2.1 Find the best exemplar for i: $s^* \leftarrow \max_k (s_{ik} + a_{ik})$, $A_i^* \leftarrow \underset{k}{\operatorname{argmax}} (s_{ik} + a_{ik})$ (can be a set of items)
   - 2.2 for all k update responsibilities

$$r_{ik} \leftarrow \begin{cases} s_{ik} - s^*, & \text{if } k \notin A_i^* \\ s_{ik} - \max_{k' \notin A_i^*} (s_{ik} + a_{ik}) & \text{otherwise} \end{cases}$$

3. for all k update availabilities
   - 3.1 $a_{kk} \leftarrow \sum_{i \neq k} [r_{ik}]_+$ where $[r_{ik}]_+ = r_{ik}$ if $r_{ik} > 0$ and 0 otherwise.
   - 3.2 for all i, $a_{ik} \leftarrow \min\{0, r_{kk} + \sum_{i' \neq i,k} [r_{i'k}]_+\}$
4. Assign an exemplar to i by $k(i) \leftarrow \underset{k'}{\operatorname{argmax}} (r_{ik'} + a_{ik'})$

Cluster validation

- External
  - when the true clustering Δ^* is known
  - compares result(s) Δ obtained by algorithm **A** with Δ^*
  - validates algorithms/methods
- Internal - no external reference

External cluster validation

Scenarios

- given data $\mathcal{D}$, truth Δ^*; algorithm **A** produces Δ
  is Δ close to Δ^*?
- given data $\mathcal{D}$, truth Δ^*; algorithm **A** produces Δ, algorithm **A'** produces Δ'
  which of Δ, Δ' is closer to Δ^*?
- multiple datasets, multiple algorithms
  which algorithm is better?

A **distance between clusterings** $d(\Delta, \Delta')$ needed

Requirements for a distance

Depend on the application

- Applies to any two partitions of the same data set
- Makes no assumptions about how the clusterings are obtained
- Values of the distance between two pairs of clusterings comparable under the weakest possible assumptions
- Metric (triangle inequality) desirable
- **understandable, interpretable**

The confusion matrix

- Let $\Delta = \{C_{1:K}\},\ \Delta' = \{C'_{1:K'}\}$
- Define $n_k = |C_k|,\ n'_{k'} = |C'_{k'}|$
- $m_{kk'} = |C_k \cap C'_{k'}|,\ k = 1:K, k' = 1:K'$
- note: $\sum_k m_{kk'} = n'_{k'},\ \sum_{k'} m_{kk'} = n_k,\ \sum_{k,k'} m_{kk'} = n$
- The **confusion matrix** $M \in \mathbb{R}^{K \times K'}$ is

$$M = [m_{kk'}]_{k=1:K}^{k'=1:K'}$$

- all distances and comparison criteria are based on M
- the **normalized confusion matrix** $P = M/n$

$$p_{kk'} = \frac{m_{kk'}}{n}$$

- The **normalized cluster sizes** $p_k = n_k/n,\ p'_{k'} = n'_{k'}/n$ are the **marginals** of P

$$p_k = \sum_{k'} p_{kk'} \quad p_{k'} = \sum_{k} p_{kk'}$$

The Misclassification Error (ME) distance

- Define the **Misclassification Error (ME)** distance d_{ME}

$$d_{ME} = 1 - \max_{\pi} \sum_{k=1}^{K} p_{k,\pi(k)} \quad \pi \in \{\text{all } K-\text{permutations}\},\ K \leq K' \text{w.l.o.g}$$

- Interpretation: treat the clusterings as classifications, then minimize the classification error over all possible label matchings
- Or: nd_{ME} is the Hamming distance between the vectors of labels, minimized over all possible label matchings
- can be computed in polynomial time by **Max bipartite matching** algorithm (also known as Hungarian algorithm)
- Is a metric: symmetric, ≥ 0, triangle inequality

$$d_{ME}(\Delta_1, \Delta_2) + d_{ME}(\Delta_1, \Delta_3) \geq d_{ME}(\Delta_2, \Delta_3)$$

- easy to understand (very popular in computer science)
- $d_{ME} \leq 1 - 1/K$
- bad: if clusterings not similar, or K large, d_{ME} is coarse/indiscriminative
- recommended: for small K

The Variation of Information (VI) distance
Clusterings as random variables

- Imagine points in $\mathcal{D}$ are picked randomly, with equal probabilities
- Then $k(i), k'(j)$ are random variables
  with $Pr[k] = p_k$, $Pr[k, k'] = p_{kk'}$

Incursion in information theory I

- **Entropy** of a random variable/clustering $H_\Delta = -\sum_k p_k \ln p_k$
- $0 \leq H_\Delta \leq \ln K$
- Measures uncertainty in a distribution (amount of randomness)
- **Joint entropy** of two clusterings

$$H_{\Delta,\Delta'} = -\sum_{k,k'} p_{kk'} \ln p_{kk'}$$

- $H_{\Delta',\Delta} \leq H_\Delta + H_{\Delta'}$ with equality when the two random variables are independent
- **Conditional entropy** of Δ' given Δ

$$H_{\Delta'|\Delta} = -\sum_k p_k \sum_{k'} \frac{p_{kk'}}{p_k} \ln \frac{p_{kk'}}{p_k}$$

- Measures the expected uncertainty about k' when k is known
- $H_{\Delta'|\Delta} \leq H_{\Delta'}$ with equality when the two random variables are independent
- **Mutual information** between two clusterings (or random variables)

$$\begin{aligned} I_{\Delta,\Delta} &= H_\Delta + H_{\Delta'} - H_{\Delta',\Delta} \\ &= H_{\Delta'} - H_{\Delta'|\Delta} \end{aligned}$$

- Measures the amount of information of one r.v. about the other
- $I_{\Delta,\Delta} \geq 0$, symmetric. Equality iff r.v.'s independent

The VI distance

- Define the **Variation of Information (VI)** distance

$$\begin{aligned} d_{VI}(\Delta, \Delta') &= H_\Delta + H_{\Delta'} - 2I_{\Delta',\Delta} \\ &= H_{\Delta|\Delta'} + H_{\Delta'|\Delta} \end{aligned}$$

- Interpretation: d_{VI} is the sum of information gained and information lost when labels are switched from $k()$ to $k'()$
- d_{VI} symmetric, ≥ 0
- d_{VI} obeys triangle inequality (is a metric)

Other properties

- Upper bound
  $d_{VI} \leq 2 \ln K_{max}$ if $K, K' \leq K_{max} \leq \sqrt{n}$
  (asymptotically attained)
- $d_{VI} \leq \ln n$ over all partitions (attained)
- Unbounded! and grows fast for small K

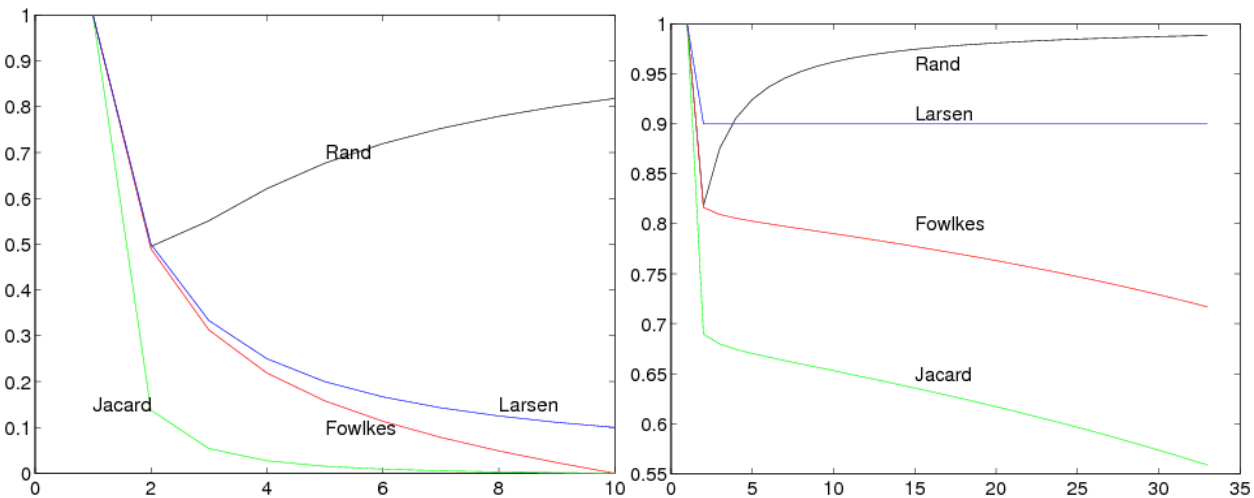
Other criteria and desirable properties

- Comparing clustering by **indices of similarity** $i(\Delta, \Delta')$
  - from statistics (Rand, adjusted Rand, Jaccard, Fowlkes-Mallows ...)
  - range=[0,1], with $i(\Delta, \Delta') = 1$ for $\Delta = \Delta'$
  - the properties of these indices not so good
  - any index can be transformed into a "distance" by $d(\Delta, \Delta') = 1 - i(\Delta, \Delta')$
- Other desirable properties of indices and distances between clusterings
  - n-invariance
  - locality
  - convex additivity

- Define $N_{11} = \#$ pairs which are together in both clusterings, $N_{12} = \#$ pairs together in Δ, separated in Δ', N_{21} (conversely), $N_{22} =$#number pairs separated in both clusterings
- Rand index $= \frac{N_{11}+N_{22}}{\#\text{pairs}}$
- Jaccard index $= \frac{N_{11}}{\#\text{pairs}}$
- Fowlkes-Mallows $=$ Precision$\times$ Recall

- all vary strongly with K. Thereforek, **Adjusted** indices used mostly

$$adj(i) = \frac{i - \bar{i}}{\max(i) - \bar{i}}$$



Internal cluster(ing) validation

Why?

- Most algorithms output a clustering even if no clusters in data (parametric algorithms)

  How to decide whether to accept it or not?
- related to selection of K
- Some algorithms are run multiple times (e.g EM)

  How to select the clustering(s) to keep?
- Validate by the cost $\mathcal{L}$
  - Δ is valid if $\mathcal{L}(\Delta)$ is "small"
- but how small is "small"?
- Note: rescaling data may change $\mathcal{L}(\Delta)$

Heuristics

- **Gap** heuristic
- single linkage:
  - define l_r length of r-th edge added to MST

$$\underbrace{l_1 \leq l_2 \leq \ldots l_{n-K}}_{\text{intracluster}} \leq \underbrace{l_{n-K+1} \leq \ldots}_{\text{deleted}}$$

  - $l_{n-K}/l_{n-K+1} \leq 1$ should be small
- min diameter:

$$\frac{\mathcal{L}(\Delta)}{\max_{i,j \in \mathcal{D}} ||x_i - x_j||}$$

$$\frac{\mathcal{L}(\Delta)}{\min_{k,k'} \text{distance}(C_k, C_{k'})}$$

- etc

Quadratic cost

- $\mathcal{L}(\Delta) = const - \operatorname{trace} X^T(\Delta) A X(\Delta)$
- with $X =$ matrix reprentation for Δ
- then, if cost value $\mathcal{L}(\Delta)$ small, we can prove that clustering Δ is almost optimal
- This holds for K-means (weighted, kernelized) and several graph partioning costs (normalized cut, average association, correlation clustering, etc)

Matrix Representations

- matrix reprentations for Δ
  - unnormalized (redundant) representation

$$\tilde{X}_{ik} = \begin{cases} 1 & i \in C_k \\ 0 & i \notin C_k \end{cases} \qquad \text{for } i = 1:n, k = 1:K$$

  - normalized (redundant) representation

$$X_{ik} = \begin{cases} 1/\sqrt{|C_k|} & i \in C_k \\ 0 & i \notin C_k \end{cases} \qquad \text{for } i = 1:n, k = 1:K$$

  therefore $X_k^T X_{k'} = \delta(k, k')$, X orthogonal matrix
  $X_k =$ column k of X
- normalized non-redundant reprentation
  - X_K is determined by $X_{1:K-1}$
  - hence we can use $Y \in \mathbb{R}^{n \times (K-1)}$ orthogonal representation
  - intuition: Y represents a subspace (is an orthogonal basis)
  - K centers in $\mathbb{R}^d$, $d \geq K$ determine a $K-1$ dimesional subspace plus a translation

- Example: the K-means cost
  - remember
$$\mathcal{L}(\Delta) = \sum_{k=1}^{K} \sum_{i,j \in C_k} \frac{1}{2|C_k|} ||x_i - x_j||^2 + \text{constant}$$
  - in matrix form
$$\mathcal{L}(\Delta) = -\frac{1}{2} X^T A X + \text{constant}$$
    where
$$A_{ij} = x_i^T x_j$$
    is the Gram matrix of the data
  - if data centered, ie $\sum_i x_i = 0$ and Y rotated appropriately (see Meila, 2006)[Meilă, 2006]
$$\mathcal{L}(\Delta) = -\frac{1}{2} Y^T A Y + \text{constant}$$
- Assume k-means cost from now on

A spectral lower bound

- minimizing $\mathcal{L}(\Delta)$ is equivalent to

$$\max Y^T A Y$$

  over all $Y \in \mathbb{R}^{n \times (K-1)}$ that represent a clustering

- a relaxation

$$\max Y^T A Y$$

  over all $Y \in \mathbb{R}^{n \times (K-1)}$ orthogonal

- solution to relaxed provlem is

$$Y^* = \text{eigenvectors}_{1:K-1} \text{ of } A$$

$$\mathcal{L}^* = \sum_{k=1}^{K-1} \lambda_k(A)$$

- $\mathcal{L}^* = \textit{constant} - L^* = \operatorname{trace} A - L^*$ is lower bound for $\mathcal{L}$

$$\mathcal{L}^* \leq \mathcal{L}(\Delta) \text{ for all } \Delta$$

A theorem (Meila, 2006)

Theorem

- define

$$\delta = \frac{Y^T A Y - \sum_{k=1}^{K-1} \lambda_k}{\lambda_{K-1} - \lambda_K} \qquad \varepsilon(\delta) = 2\delta[1 - \delta/(K-1)]$$

- define $p_{min}, p_{max} = \frac{\min,\max |C_k|}{n}$
- then, whenever $\varepsilon(\delta) \leq p_{min}$, we have that

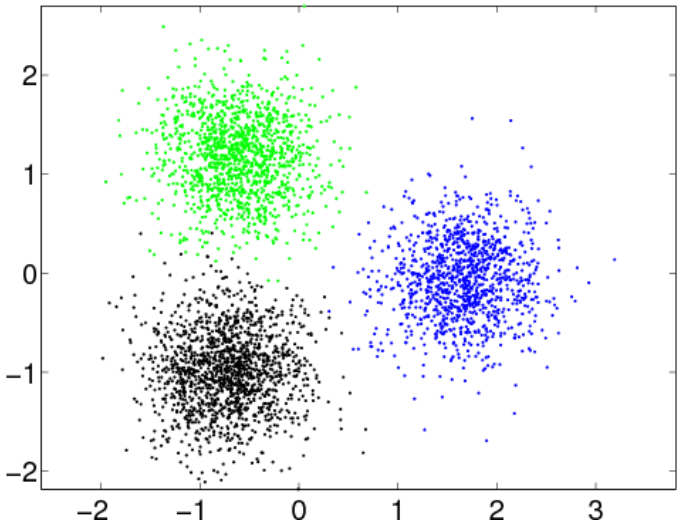
$$d_{ME}(\Delta, \Delta^{opt}) \leq \varepsilon(\delta) p_{max}$$

where d_{ME} is misclassification error distance

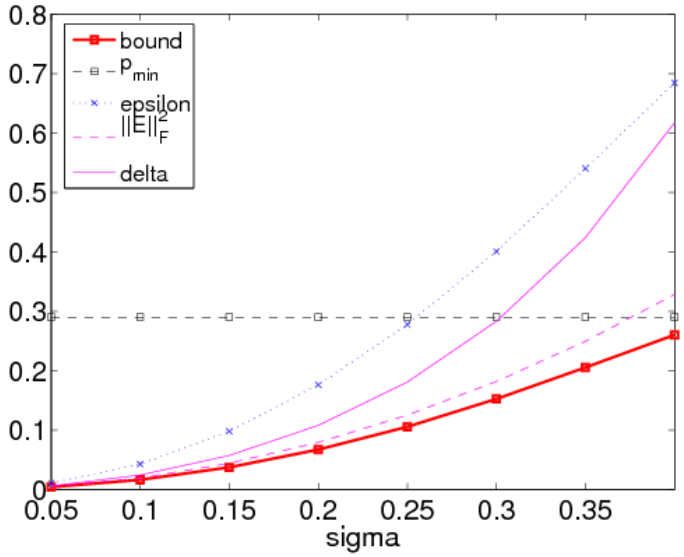
Remarks

- it is a worst-case result
- makes no (implicit) distributional assumptions
- when theorem applies, bound is good $d_{ME}(\Delta, \Delta^{opt}) \leq p_{min}$
- applies only if a good clustering is found (not all data, clusterings)
- intuiton: if data well clustered, $K-1$ principal subspace is aligned with cluster centers

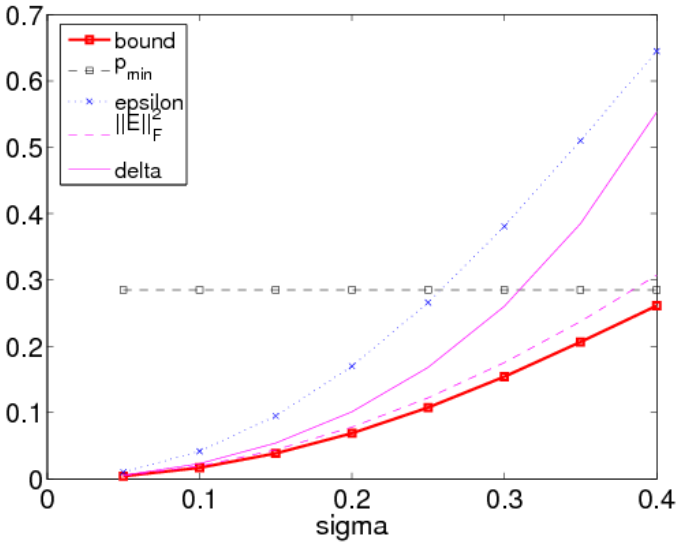
data $d = 35$, $\sigma = 0.4$



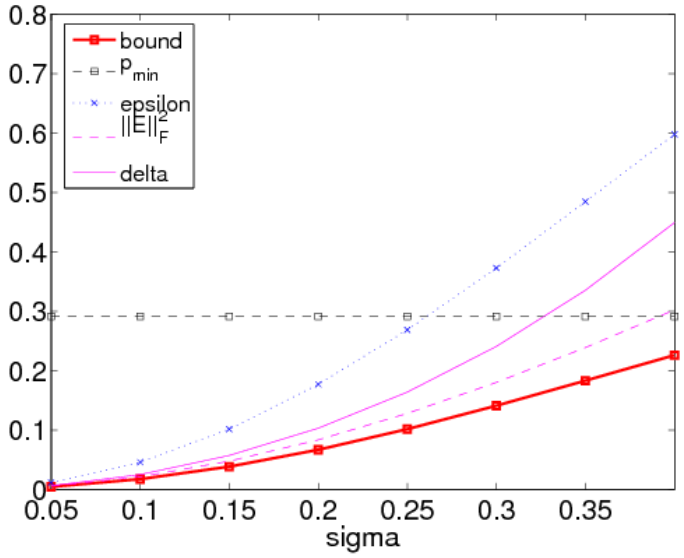
$n = 100$



$n = 200$



$n = 1000$



What I didn't talk about

- Hierarchical clustering
- Subspace clustering (or clustering on subsets of attributes)
- Bi-clustering (and multi-way-clustering)
- Partial clustering
- Ensembles of clusterings, consensus clustering, and clustering clusterings

Hierarchical clustering

- **Divisive** (top down)
  - starts with all data in one cluster, divides recursively into 2 (or more) clusters
  - Example: spectral clustering, min diameter
- **Agglomerative** (bottom up)
  - starts n cluster containing 1 item, merges 2 clusters recursively
  - Example: Ward algorithm, single linkage
- **Hierarchical Dirichlet processes**
- Remarks
  - Any cost based clustering paradigm can produce a hierarchical clustering
  - Any non-parametric level-sets paradigm can produce a hierarchical clustering
  - Mixture models (finite or not) can also be defined hierarchically. Issues of identifyability appear

The Ward agglomerative algoritm [Ward, 1963]

- Cost $=$ same as K-means
- Algorithm idea:
  - Start with n single point clusters
  - Merge the two clusters that increase $\mathcal{L}$ the least, until K clusters left
- **Greedy**, recursive algorithm, $\mathcal{O}(n^3)$ operations

Subspace clustering

- Problem: each cluster is defined by a subset of relevant attributes (features)
  - Examples: user modeling (clusters of users vs clusters of products/services), gene expression data
- Known as **Clustering on Subsets of Attributes (COSA) Biclustering (and Multiway Clustering), Subspace clustering**
- Amounts to clustering both the data exemplars and the data features
- Approaches
  - **COSA** [Friedman and Meulman, 2004] cost based, + additional entropy term. Alternate minimization algorithm.
  - **[?]** Dirichlet process mixtures approach. Each $f(.;\theta_k)$ samples a set of relevant features. Estimated by MCMC
  - **Multivariate Information Bottleneck** [Friedman et al., 2001] Information theory based. Estimation by alternate (KL-divergence) projections.
  - many others. . . see IEEE TKDE

Partial clustering

- Problem: Given a node, find its cluster
- Premise: the data set is extremely large, there are many small clusters, possibly $\mathcal{O}(n)$
- **Nibble** algorithm of [Spielman and Teng, 2008]

  Given: a graph, by its Markov transition matrix P
  Start with node i, tolerance ε, number steps t
  Initialize $p \in \mathbb{R}^n$ with $p_i = 1$, $p_j = 0$ for $j \neq i$
  - Iterate for t steps
    1. $p \leftarrow Pp$
    2. for $j = 1 : n$, if $p_j < \varepsilon$ set $p_j = 0$

  Output $C(i) = \{ j \mid p_j > 0 \}$
- $C(i)$ is the set of items attainable from i by a "likely" path
- Original algorithm has sparsest cut guarantees
  Used as subroutine by other algorithms.

Links

- Yee Whye Teh's tutorial on DP Mixtures http://mlg.eng.cam.ac.uk/tutorials/07/ywt.pdf
- Lecture on exponential family models http:


Achlioptas, D. and McSherry, F. (2005).
On spectral learning of mixtures of distributions.
In Auer, P. and Meir, R., editors, *18th Annual Conference on Learning Theory, COLT 2005*, pages 458–471, Berlin/Heidelberg. Springer.


Arora, S. and Kannan, R. (2001).
Learning mixtures of arbitrary gaussians.
In *STOC '01: Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 247–257, New York, NY, USA. ACM Press.


Azran, A. and Ghahramani, Z. (2006).
Spectral methods for automatic multiscale data clustering.
In *Computer Vision and Pattern Recognition*, pages 190–197. IEEE Computer Society.


Bach, F. and Jordan, M. I. (2006).
Learning spectral clustering with applications to speech separation.
Journal of Machine Learning Research, 7:1963–2001.


Banfield, J. D. and Raftery, A. E. (1993).
Model-based gaussian and non-gaussian clustering.
Biometrics, 49:803–821.


Ben-Hur, A., Elisseeff, A., and Guyon, I. (2002).
A stability based method for discovering structure in clustered data.
In *Pacific Symposium on Biocomputing*, pages 6–17.


Bradley, P. and Mangasarian, O. (2005).
Clustering via concave minimization.

In *Advances in Neural Information Processing systems (NIPS)*, Cambridge, MA. MIT Press.


Bubeck, S., Meilă, M., and von Luxburg, U. (2009).
How the initialization affects the stability of the k-means algorithm.
Technical Report arXiv:0907.5494v1 [stat.ML], ArXiv.


Carreira-Perpinan, M. A. (2007).
Gaussian mean shift is an EM algorithm.
IEEE Trans. on Pattern Analysis and Machine Intelligence, 29(5):767–776.


Charikar, M. and Guha, S. (1999).
Improved combinatorial algorithms for the facility location and k-median problems.
In *40th Annual Symposium on Foundations of Computer Science*, pages 378–388.


Dasgupta, S. (2000).
Experiments with random projection.
In *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 143–151, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.


Dasgupta, S. and Gupta, A. (2002).
An elementary proof of a theorem of johnson and lindenstrauss.
Algorithms, 22:60–65.


Dasgupta, S. and Schulman, L. (2007).
A probabilistic analysis of em for mixtures of separated, spherical gaussians.
Journal of Machine Learnig Research, 8:203–226.


Frey, B. J. and Dueck, D. (2007).

Clustering by passing messages between data points.
Science, 315:973–976.


Friedman, J. and Meulman, L. (2004).
Clustering on subsets of attributes.
Journal of the Royal Statistical Society B.


Friedman, N., Mosenzon, O., Slonim, N., and Tishby, N. (2001).
Multivariate information bottleneck.
In *Uncertainty in Artificial Intelligence: Proceedings of the Seventeenth Conference (UAI-2001)*, pages 152–161, San Francisco, CA. Morgan Kaufmann Publishers.


Har-Peled, S. and Mazumdar, S. (2004).
Coresets for k-means and k-median clustering and their applications.
In *Proc. 36th Annu. ACM Sympos. Theory Comput (STOC)*, pages 291–300.


Hochbaum, D. S. and Shmoys, D. B. (1985).
A best possible heuristic for the k-center problem.
Mathematics of Operations Research, 10(2):180–184.


Lange, T., Roth, V., Braun, M. L., and Buhmann, J. M. (2004).
Stability-based validation of clustering solutions.
Neural Comput., 16(6):1299–1323.


Lloyd, S. P. (1982).
Least squares quantization in PCM.
IEEE Transactions on Information Theory, 28:129–137.


McLachlan, G. J. and Krishnan, T. (1997).
The EM algorithm and extensions.

Wiley, New York, NY.


Meilă, M. (2006).
The uniqueness of a good optimum for K-means.
In Moore, A. and Cohen, W., editors, *Proceedings of the International Machine Learning Conference (ICML)*, pages 625–632. International Machine Learning Society.


Meilă, M. and Shi, J. (2001a).
Learning segmentation by random walks.
In Leen, T. K., Dietterich, T. G., and Tresp, V., editors, *Advances in Neural Information Processing Systems*, volume 13, pages 873–879, Cambridge, MA. MIT Press.


Meilă, M. and Shi, J. (2001b).
A random walks view of spectral segmentation.
In Jaakkola, T. and Richardson, T., editors, *Artificial Intelligence and Statistics AISTATS*.


Meilă, M., Shortreed, S., and Xu, L. (2005).
Regularized spectral learning.
In Cowell, R. and Ghahramani, Z., editors, *Proceedings of the Artificial Intelligence and Statistics Workshop(AISTATS 05)*.


Nadler, B., Lafon, S., Coifman, R., and Kevrekidis, I. (2006).
Diffusion maps, spectral clustering and eigenfunctions of fokker-planck operators.
In Weiss, Y., Schölkopf, B., and Platt, J., editors, *Advances in Neural Information Processing Systems 18*, pages 955–962, Cambridge, MA. MIT Press.


Neal, R. M. and Hinton, G. E. (1998).

A view of the em algorithm that justifies incremental, sparse, and other variants.
In Jordan, M. I., editor, *Learning in Graphical Models*, NATO Science series, pages 355–368. Kluwer Academic Publishers.

Ng, A. Y., Jordan, M. I., and Weiss, Y. (2002).
On spectral clustering: Analysis and an algorithm.
In Dietterich, T. G., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA. MIT Press.

Nugent, R. and Meila, M. (2010).
Statistical Methods in Molecular Biology, chapter An Overview of Clustering Applied to Molecular Biology.
Humana Press, Springer.

Pelleg, D. and Moore, A. (2000).
X-means: Extending K-means with efficient estimation of the number of clusters.
In Bratko, I. and Džeroski, S., editors, *Proceedings of the 17th International Conference on Machine Learning*, pages 727–734, San Francisco, CA. Morgan Kaufmann.

Shortreed, S. and Meilă, M. (2005).
Unsupervised spectral learning.
In Jaakkola, T. and Bachhus, F., editors, *Proceedings of the 21st Conference on Uncertainty in AI*, pages 534–544, Arlington,Virginia. AUAI Press.

Spielman, D. and Teng, S.-H. (2008).
A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning.
Technical Report :0809.3232v1 [cs.DS], arXiv.


Srebro, N., Shakhnarovich, G., and Roweis, S. (2006).
An investigation of computational and informational limits in gaussian mixture clustering.
In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*.


Tishby, N. and Slonim, N. (2000).
Data clustering by markovian relaxation and the information bottleneck method.
In *NIPS*, pages 640–646.


Vempala, S. and Wang, G. (2004).
A spectral algorithm for learning mixtures of distributions.
Journal of Computer Systems Science, 68(4):841–860.


Verma, D. and Meilă, M. (2003).
A comparison of spectral clustering algorithms.
TR 03-05-01, University of Washington.


von Luxburg, U. (2009).
Clustering stability: An overview.
Foundations and Trends in Machine Learning, 2(3):253–274.


Ward, J. H. (1963).
Hierarchical grouping to optimize an objective function.
Journal of American Statistical Association, 58(301):236–244.