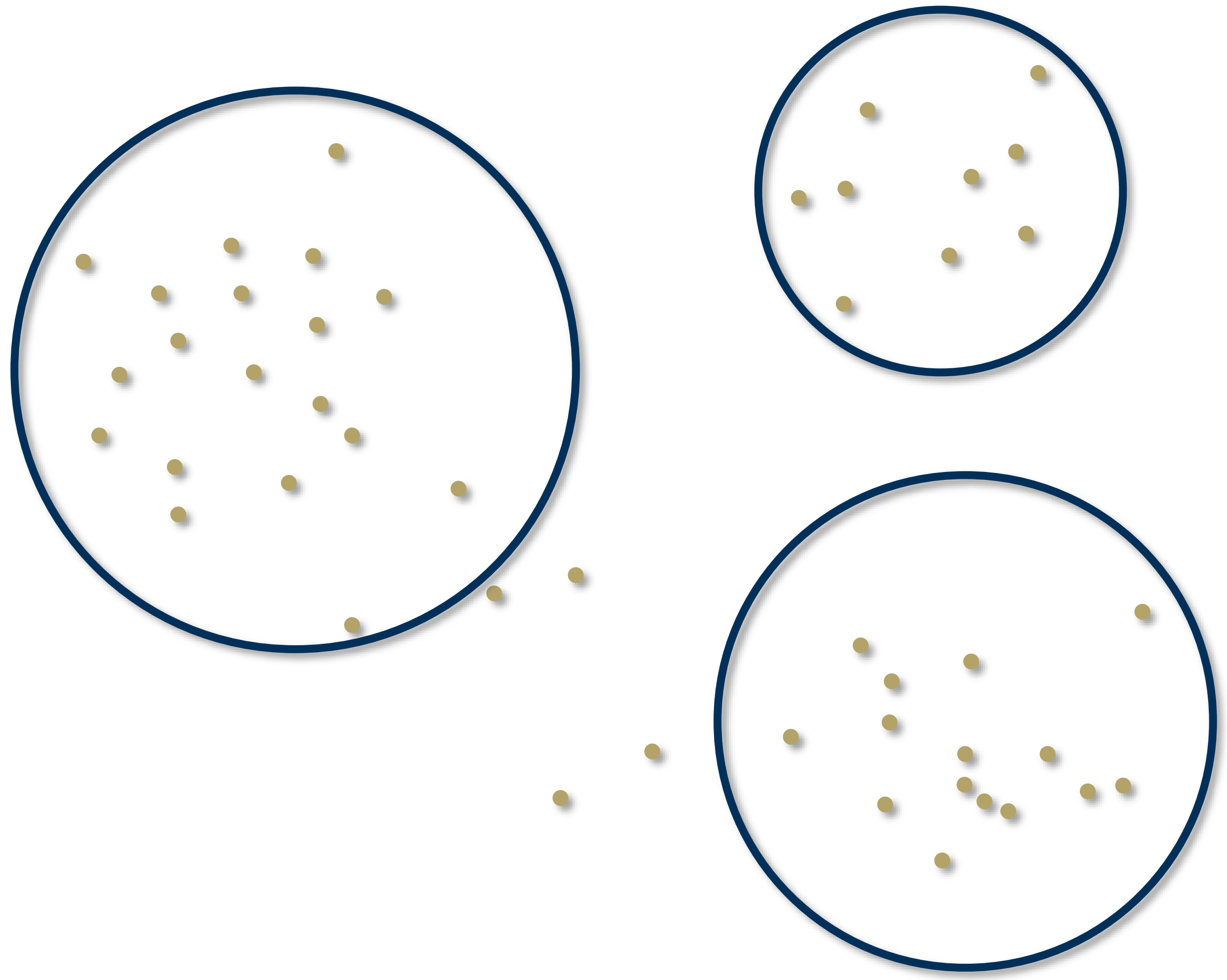
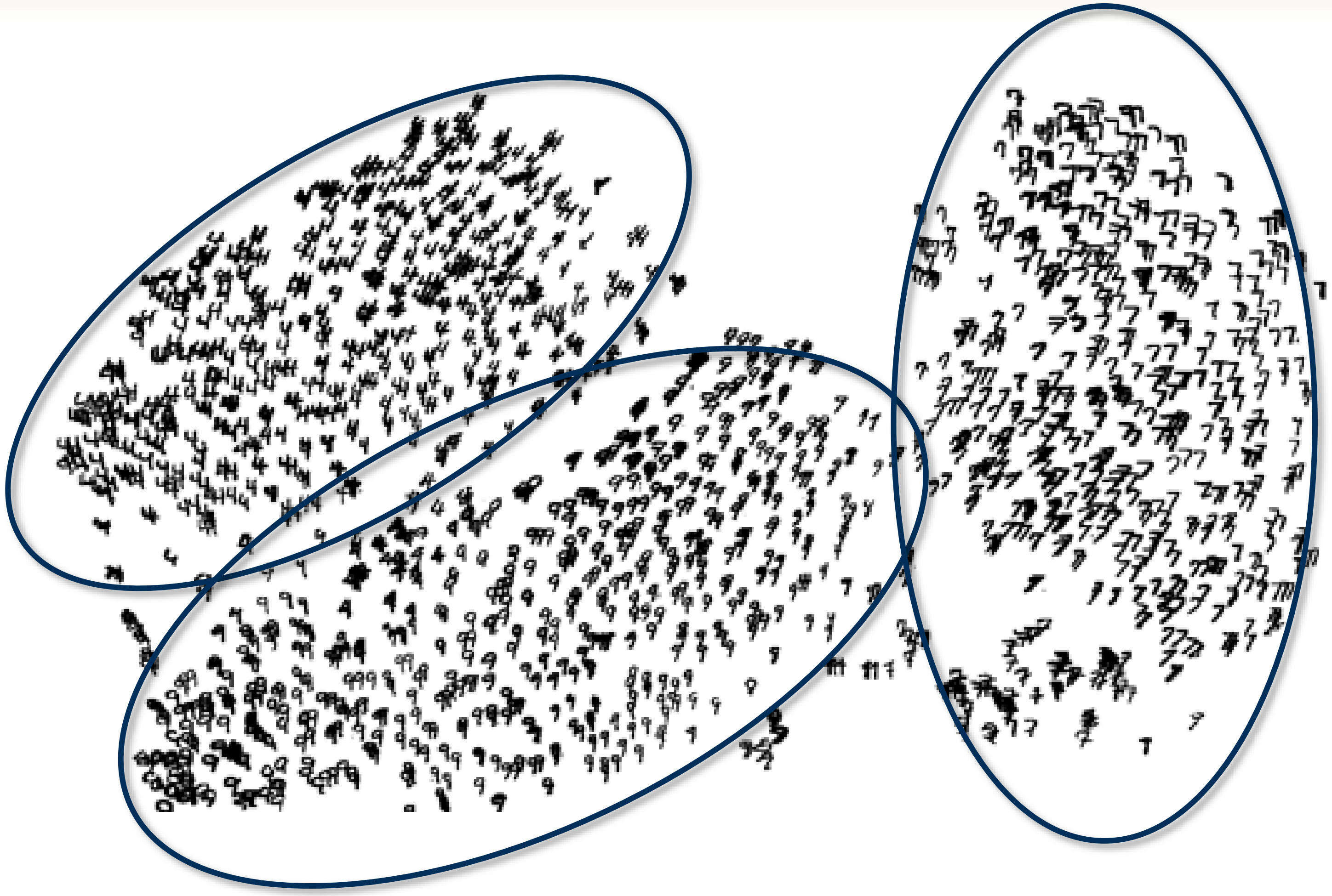
Clustering



Example



Formal definition

Suppose $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$

The goal of ***clustering*** is to assign the data to disjoint subsets called ***clusters***, so that points in the same cluster are more similar to each other than points in different clusters

A clustering can be represented by a cluster map, which is a function

$$C : \{1, \ldots, n\} \to \{1, \ldots, K\}$$

where K is the number of clusters

K-means criterion

Choose C to minimize

$$W(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_2^2$$

where

$$\boldsymbol{\mu}_k := \frac{1}{n_k} \sum_{i:C(i)=k} \mathbf{x}_i \qquad n_k = |\{i : C(i) = k\}|$$

Note that K is assumed fixed and known

$W(C)$ is sometimes called the "***within-cluster scatter***"

Minimizing the K-means criterion

There is no known efficient search strategy for this space

Can be solved (exactly) in time $O(n^{dK+1} \log n)$

Completely impractical unless both d and K are extremely small

- e.g., $d = 2, K = 3$ already results in $O(n^7 \log n)$

More formally, minimizing the K-means criterion is a ***combinatorial*** optimization problem (NP-hard)

Instead, we resort to an ***iterative, suboptimal algorithm***

Another look at K-means

Recall that we want to find

$$C^* = \arg\min_{C} \sum_{k=1}^{K} \sum_{i:C(i)=k} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_2^2$$

Note that for fixed C

$$\boldsymbol{\mu}_k = \arg\min_{\mathbf{m}} \sum_{i:C(i)=k} \|\mathbf{x}_i - \mathbf{m}\|_2^2$$

Therefore, we can equivalently write

$$C^* = \arg\min_{C,\{\mathbf{m}_k\}_{k=1}^{K}} \sum_{k=1}^{K} \sum_{i:C(i)=k} \|\mathbf{x}_i - \mathbf{m}_k\|_2^2$$

An iterative algorithm

$$C^* = \arg\min_{C, \{\mathbf{m}_k\}_{k=1}^K} \underbrace{\sum_{k=1}^{K} \sum_{i:C(i)=k} \|\mathbf{x}_i - \mathbf{m}_k\|_2^2}_{W(C, \{\mathbf{m}_k\}_{k=1}^K)}$$

This suggests an iterative algorithm

1. Given C, choose $\mathbf{m}_k$ to minimize $W(C, \{\mathbf{m}_k\}_{k=1}^K)$

2. Given $\mathbf{m}_k$, choose C to minimize $W(C, \{\mathbf{m}_k\}_{k=1}^K)$

K-means clustering algorithm

The solutions to each sub-problem are given by

1. $\mathbf{m}_k^* = \frac{1}{n_k} \sum_{i:C(i)=k} \mathbf{x}_i$

2. $C^*(i) = \arg\min_k \|\mathbf{x}_i - \mathbf{m}_k\|_2^2$

Algorithm

Initialize $\mathbf{m}_k, k = 1, \ldots, K$

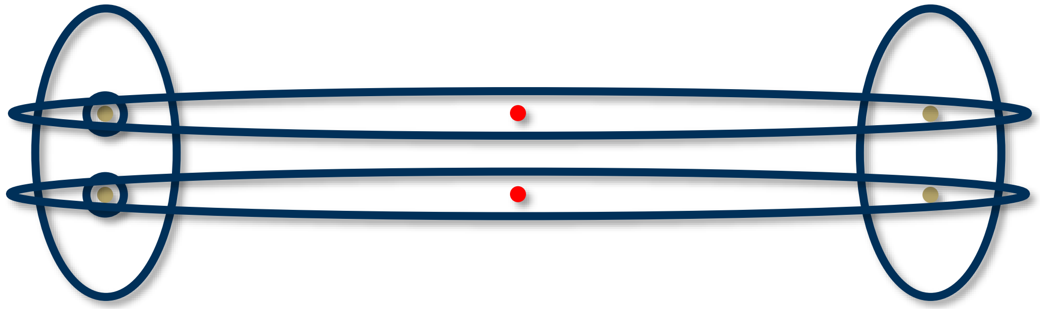
Repeat until clusters don't change

- $C(i) = \arg\min_k \|\mathbf{x}_i - \mathbf{m}_k\|_2^2$
- $\mathbf{m}_k = \frac{1}{n_k} \sum_{i:C(i)=k} \mathbf{x}_i$

Initialization

Traditionally, the algorithm is typically initialized by setting each $\mathbf{m}_k$ to be a ***random*** point in the dataset

However, depending on the initialization, the algorithm can get stuck in a local minimum



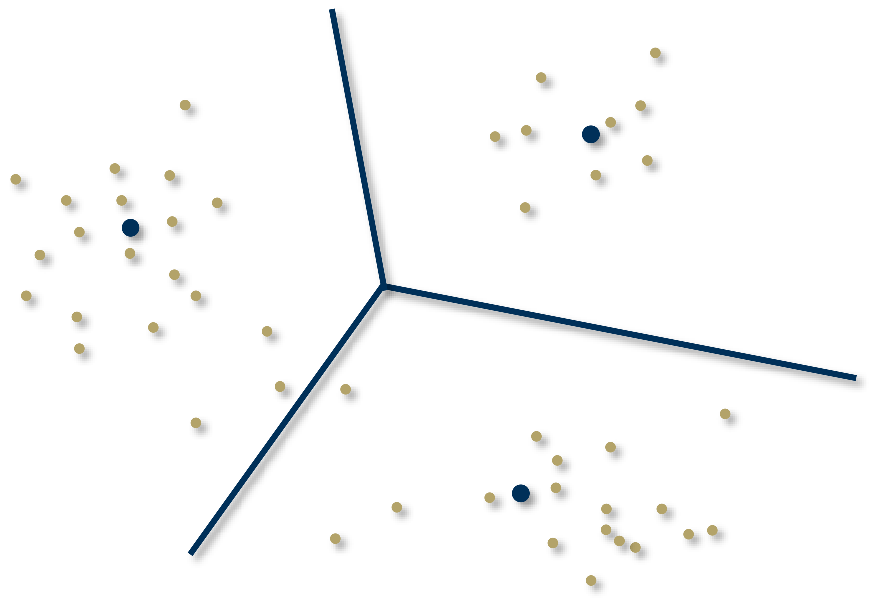
One can avoid this by:

- repeating for several random initializations
- initialize by ***sequentially*** selecting random points in the dataset, but with a probability depending on how far the point is from the already selected $\mathbf{m}_k$: *K-means ++*

Cluster geometry

Clusters are "nearest neighbor" regions or ***Vornoi cells*** defined with respect to the cluster means

Cluster boundaries are formed by the intersections of ***hyperplanes***



K-means will "fail" if clusters are ***nonconvex***

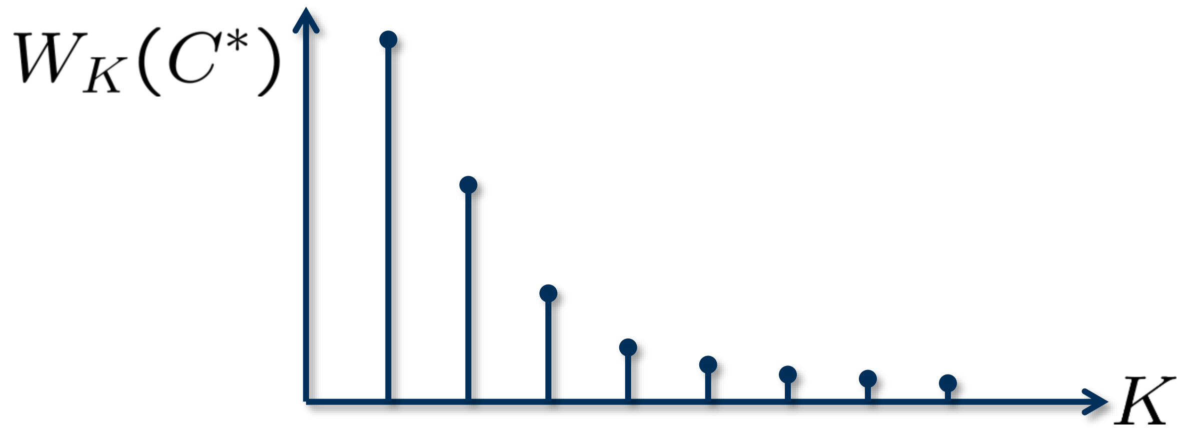
Remarks

- Algorithm originally developed at Bell Labs as an approach to vector quantization
- If we replace the ℓ_2 norm with the ℓ_1 norm in our function $W(C)$, then
  - the geometry of our Vornoi regions will change
  - the "center" of each region is actually calculated via the median in each dimension
  - results in *K-medians clustering*

Model selection for K-means

How to choose K ?

Let $W_K(C^*)$ be the within-cluster scatter based on K clusters



If the "right" number of clusters is K^*, we expect

- for $K < K^*$, $W_K(C^*) - W_{K-1}(C^*)$ will be ***large***
- for $K > K^*$, $W_K(C^*) - W_{K-1}(C^*)$ will be ***small***

This suggests choosing K to be near the "knee" of the curve

Another take on K-means

I have followed the standard development of the K-means clustering algorithm, but there is another way to view this algorithm…

as simply another instance of structured matrix factorization

$$\mathbf{X} \approx \mathbf{B}\mathbf{C}$$

where

$$\mathbf{B} = \begin{bmatrix} | & & | \\ \mathbf{m}_1 & \cdots & \mathbf{m}_k \\ | & & | \end{bmatrix}$$

and $\mathbf{C}$ has exactly one “1” per column (the rest being zero)

Beyond K-means clustering

In K-means clustering, by measuring the within-cluster scatter as

$$W(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|_2^2$$

we are implicitly assuming that each cluster is roughly spherical in shape

This is probably unrealistic

But if we don’t even know what the clusters are
we definitely don’t know how they are shaped…

Clustering with Gaussian mixture models

One way to extend the basic idea behind K-means clustering to allow for more general cluster shapes is to assume

- the clusters are ***elliptical***
- each cluster can be modeled using a ***multivariate Gaussian density***
- the full data set is modeled using a ***Gaussian mixture model*** (GMM)

We can then perform clustering based on a maximum likelihood estimation of the GMM

Gaussian mixture models

Recall the multivariate Gaussian density

$$\phi(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = (2\pi)^{-\frac{d}{2}}|\boldsymbol{\Sigma}|^{-\frac{1}{2}}e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$

where $\mathbf{x} \in \mathbb{R}^d, \boldsymbol{\mu} \in \mathbb{R}^d, \boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}, \boldsymbol{\Sigma} \succeq 0$

A random variable X follows a ***Gaussian mixture model*** if its density has the form

$$f(\mathbf{x}) = \sum_{k=1}^{K} w_k \phi(\mathbf{x};\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)$$

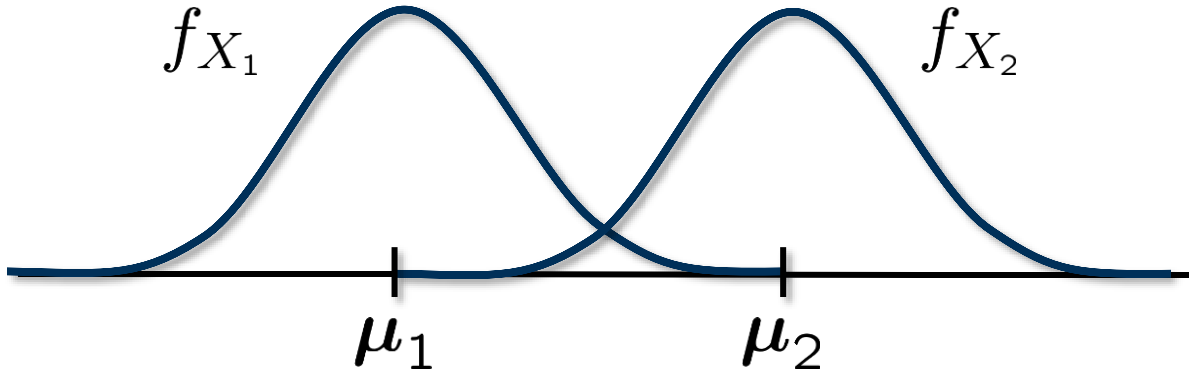
where $0 < w_k < 1, \sum_{k=1}^{K} w_k = 1$

$\boldsymbol{\mu}_k \in \mathbb{R}^d$

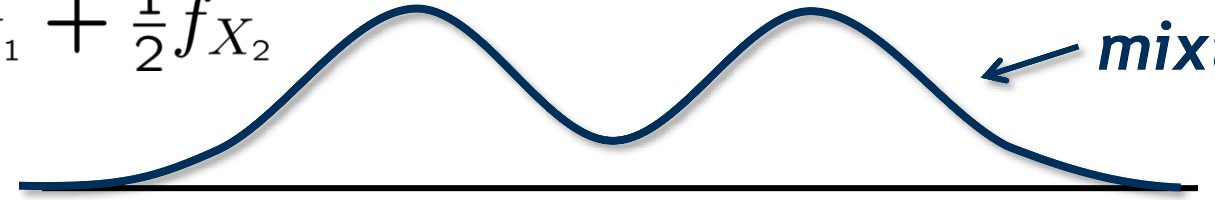
$\boldsymbol{\Sigma}_k \in \mathbb{R}^{d\times d}, \boldsymbol{\Sigma}_k \succeq 0$

Example

$X_1 \sim \mathcal{N}(\mu_1, \sigma^2)$ $X_2 \sim \mathcal{N}(\mu_2, \sigma^2)$

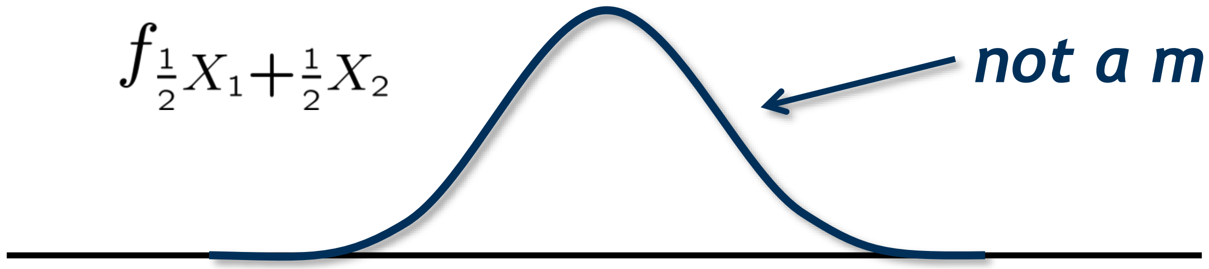


$\frac{1}{2}f_{X_1} + \frac{1}{2}f_{X_2}$



mixture

$f_{\frac{1}{2}X_1+\frac{1}{2}X_2}$



not a mixture

Simulating a GMM

Let $S \in \{1, \ldots, K\}$ be a discrete random variable such that

$$\mathbb{P}\left[S = k\right] = w_k$$

Generate X as follows:

1. Generate a realization s of S
2. Generate $X \sim \mathcal{N}(\boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s)$

The density of X generated this way is

$$\begin{aligned} f(\mathbf{x}) &= \sum_{k=1}^{K} f(\mathbf{x}|S = k) \cdot \mathbb{P}[S = k] \\ &= \sum_{k=1}^{K} w_k \phi(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \end{aligned}$$

GMMs for clustering

The variable S is called a (hidden) ***state variable***

We can imagine that every realization from a GMM is actually associated with a (hidden) realization of the state variable S

In the context of clustering, our objective is to estimate the parameters of the GMM and define clusters based on the estimated means/covariances

That is, we assume $\mathbf{x}_1, \ldots, \mathbf{x}_n \sim f(\mathbf{x}; \boldsymbol{\theta})$ and reduce clustering to a ***parameter estimation*** problem

Of course, we have to do all of this without observing the hidden states $s_1, \ldots, s_n$ associated with $\mathbf{x}_1, \ldots, \mathbf{x}_n$

MLE for a GMM

Now consider a GMM and assume that K is known

The likelihood function is

$$\mathcal{L}(\boldsymbol{\theta}; \mathbf{X}) = \prod_{i=1}^{n} f(\mathbf{x}_i; \boldsymbol{\theta}) = \prod_{i=1}^{n} \left(\sum_{k=1}^{K} w_k \phi(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

and the log likelihood is

$$\ell(\boldsymbol{\theta}; \mathbf{X}) = \sum_{i=1}^{n} \log \left(\sum_{k=1}^{K} w_k \phi(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

There is (unfortunately) no known closed-form maximizer

MLE with “incomplete data”

Define the “indicator” variable

$$\Delta_{i,k} = \begin{cases} 1 & \text{if } s_i = k \\ 0 & \text{if } s_i \neq k \end{cases}$$

The ***complete data log-likelihood*** can be written as

$$\begin{aligned} \ell(\boldsymbol{\theta}; \mathbf{X}, \mathrm{s}) &= \sum_{i=1}^{n} \log \left(\sum_{k=1}^{K} \Delta_{i,k} \cdot w_k \cdot \phi(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right) \\ &= \sum_{i=1}^{n} \sum_{k=1}^{K} \Delta_{i,k} \left(\log w_k + \log \phi(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right) \end{aligned}$$

Expectation-Maximization (EM)

The ***expectation-maximization (EM)*** algorithm is an iterative algorithm that produces a sequence $\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots$ of parameter estimates

E-step

Given $\boldsymbol{\theta}^{(j)}$, compute the ***expected complete data log-likelihood***:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(j)}) = \mathbb{E}_{\mathbf{S}|\mathbf{X}}\left[\ell(\boldsymbol{\theta}; \mathbf{X}, \mathbf{S}) | \mathbf{X}; \boldsymbol{\theta}^{(j)}\right]$$

In our case

where $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(j)}) = \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{i,k}(\boldsymbol{\theta}^{(j)}) \left(\log w_k + \log \phi(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right)$

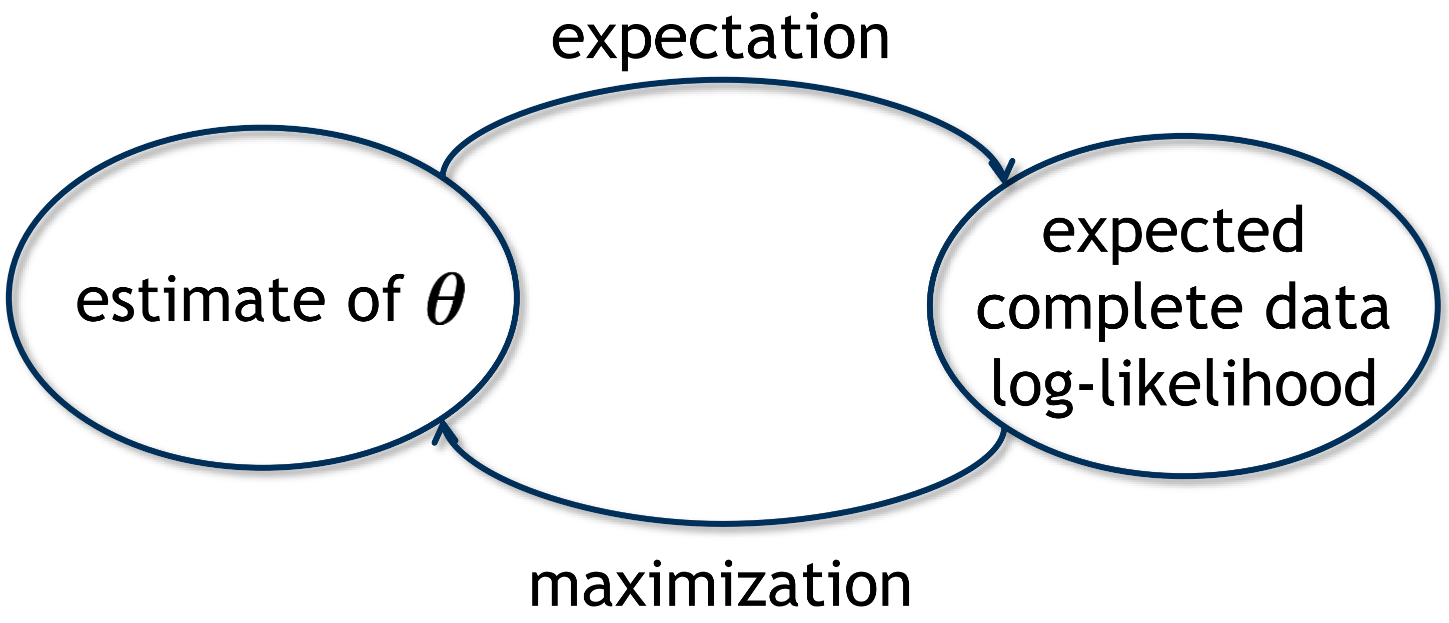
$$\gamma_{i,k}(\boldsymbol{\theta}^{(j)}) := \mathbb{E}_{\mathbf{S}|\mathbf{X}}\left[\Delta_{i,k} | \mathbf{X}; \boldsymbol{\theta}^{(j)}\right]$$

Expectation-Maximization (EM)

M-step

Given the expected complete data log-likelihood, compute the maximum likelihood estimate

$$\boldsymbol{\theta}^{(j+1)} = \arg\max_{\boldsymbol{\theta}} \; Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(j)})$$


expectation
estimate of $\boldsymbol{\theta}$
expected complete data log-likelihood
maximization

EM algorithm for GMMs

Initialize $w_k^{(0)}, \boldsymbol{\mu}_k^{(0)}, \boldsymbol{\Sigma}_k^{(0)}$

Repeat unit termination criterion satisfied

E-Step: Compute

$$\gamma_{i,k}^{(j)} = \frac{w_k^{(j)} \phi(\mathbf{x}_i; \boldsymbol{\mu}_k^{(j)}, \boldsymbol{\Sigma}_k^{(j)})}{\sum_{\ell=1}^{K} w_\ell^{(j)} \phi(\mathbf{x}_i; \boldsymbol{\mu}_\ell^{(j)}, \boldsymbol{\Sigma}_\ell^{(j)})}$$

M-Step: Compute

$$w_k^{(j+1)} = \frac{\sum_{i=1}^{n} \gamma_{i,k}^{(j)}}{n} \qquad \boldsymbol{\mu}_k^{(j+1)} = \frac{\sum_{i=1}^{n} \gamma_{i,k}^{(j)} \mathbf{x}_i}{\sum_{i=1}^{n} \gamma_{i,k}^{(j)}}$$

$$\boldsymbol{\Sigma}_k^{(j+1)} = \frac{\sum_{i=1}^{n} \gamma_{i,k}^{(j)} (\mathbf{x}_i - \widehat{\boldsymbol{\mu}}_k^{(j+1)})(\mathbf{x}_i - \widehat{\boldsymbol{\mu}}_k^{(j+1)})^T}{\sum_{i=1}^{n} \gamma_{i,k}^{(j)}}$$

Initialization and termination

In general, the likelihood has many local maxima, so a good initialization of the algorithm is critical

A good initialization for EM in the case of GMM is

$$w_k^{(0)} = \frac{1}{K}$$

$$\boldsymbol{\mu}_k^{(0)} = \text{ a randomly selected } \mathbf{x}_i$$

(sampled without replacement)

$$\boldsymbol{\Sigma}_k^{(0)} = \text{ the sample covariance}$$

Possible termination criteria are to stop iterating when

$$|\ell(\boldsymbol{\theta}^{(j+1)}; \mathbf{X}) - \ell(\boldsymbol{\theta}^{(j)}; \mathbf{X})| \leq \epsilon$$

or

$$|Q(\boldsymbol{\theta}^{(j+1)}, \boldsymbol{\theta}^{(j)}) - Q(\boldsymbol{\theta}^{(j)}, \boldsymbol{\theta}^{(j)})| \leq \epsilon$$

Defining clusters

Recall that

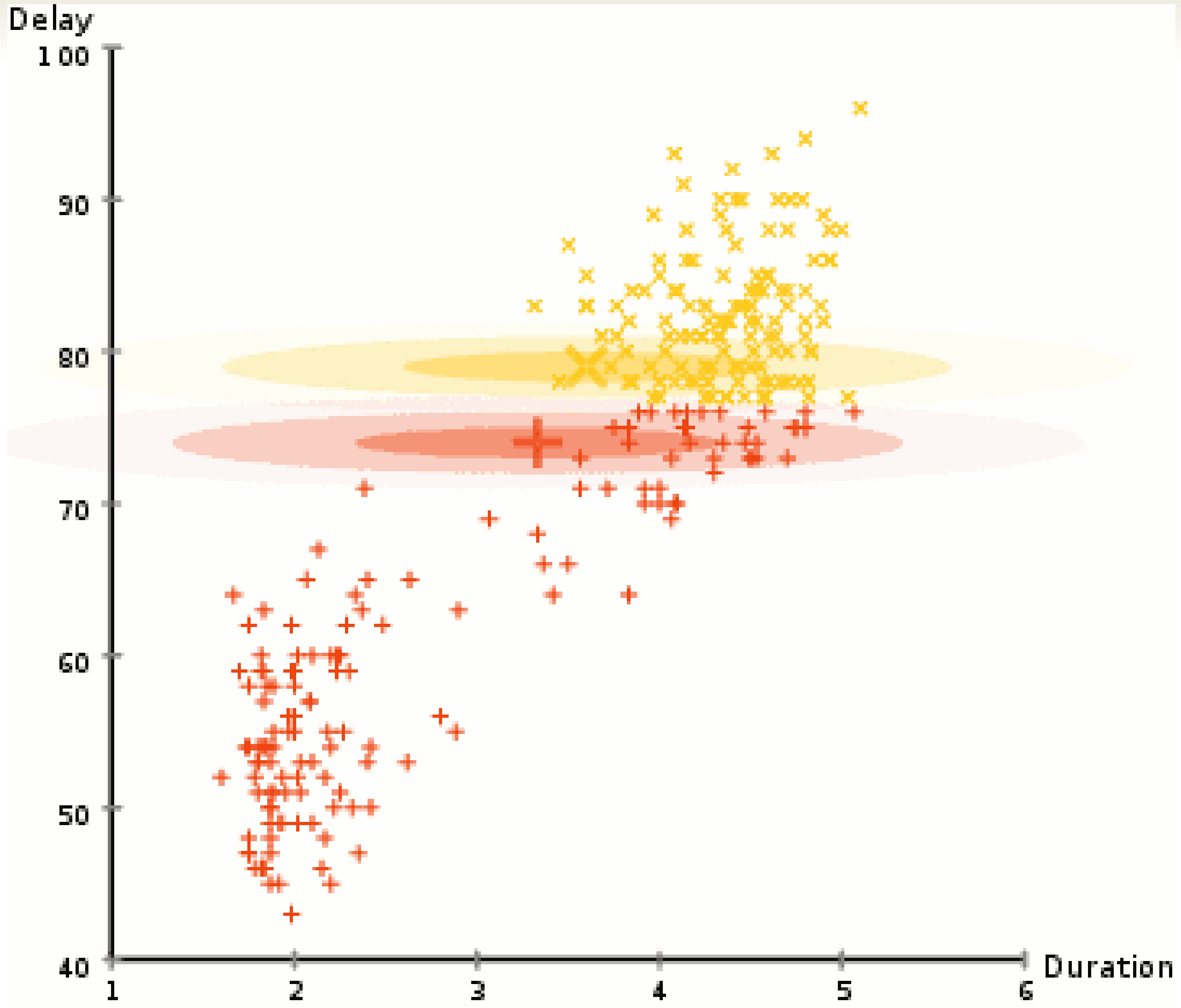
$$\gamma_{i,k}(\boldsymbol{\theta}^{(j)}) = \mathbb{P}\left[S_i = k | \mathbf{X}; \boldsymbol{\theta}^{(j)}\right]$$

A reasonable "hard" assignment of points to clusters is given by

$$C(i) = \operatorname*{arg\,max}_{k=1,\ldots,K} \gamma_{i,k}(\widehat{\boldsymbol{\theta}})$$

Alternatively, one may simply take $\gamma_{i,k}(\widehat{\boldsymbol{\theta}})$ as a "soft" assignment that expresses the "affinity" of $\mathbf{x}_i$ for cluster k

Example: Eruptions of "Old Faithful"



EM in general

Nothing in the EM algorithm as we have stated it is specific to GMMs

EM is actually an extremely general algorithm for computing ML/MAP estimators

Applies whenever having knowledge of certain “hidden variables” renders an ML/MAP estimator tractable

See *Statistical Analysis with Missing Data* by Little and Rubin (2002) for an in-depth discussion of other applications

Convergence of EM

Theorem

For each $j = 1, 2, \ldots$

$$\ell(\boldsymbol{\theta}^{(j+1)}; \mathbf{X}) \geq \ell(\boldsymbol{\theta}^{(j)}; \mathbf{X})$$

A proof based on Jensen's inequality is available in Hastie, Tibshirani, and Friedman

The convergence rate is also of interest

It is typically linear, but with a rate that depends on the proportion of observed data

Connection to K-means

Consider a GMM where each $\boldsymbol{\Sigma}_k = \sigma^2\mathbf{I}$ for some fixed σ^2

The EM algorithm for computing the MLE of $\{w_k\}_{k=1}^K$ and $\{\boldsymbol{\mu}_k\}_{k=1}^K$ is to iterate

$$\gamma_{i,k} = \frac{w_k \phi(\mathbf{x}_i; \boldsymbol{\mu}_k, \sigma^2\mathbf{I})}{\sum_{\ell=1}^K w_\ell \phi(\mathbf{x}_i; \boldsymbol{\mu}_\ell, \sigma^2\mathbf{I})}$$

$$w_k = \tfrac{1}{n}\textstyle\sum_{i=1}^n \gamma_{i,k}$$

$$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^n \gamma_{i,k}\mathbf{x}_i}{\sum_{i=1}^n \gamma_{i,k}}$$

When $\sigma^2 \to 0$, $\gamma_{i,k} = \rightarrow \begin{cases} 1 & \text{if } k = \arg\min_\ell \|\mathbf{x}_i - \boldsymbol{\mu}_\ell\|_2 \\ 0 & \text{otherwise} \end{cases}$

so the algorithm reduces to K-means