Introduction to Machine Learning CMU-10701

19. Clustering and EM

Barnabás Póczos

Contents

- ❑ Clustering
  - ❑ K-means
  - ❑ Mixture of Gaussians
- ❑ Expectation Maximization
- ❑ Variational Methods

Many of these slides are taken from

- Aarti Singh,
- Eric Xing,
- Carlos Guetrin

Clustering

What is clustering?

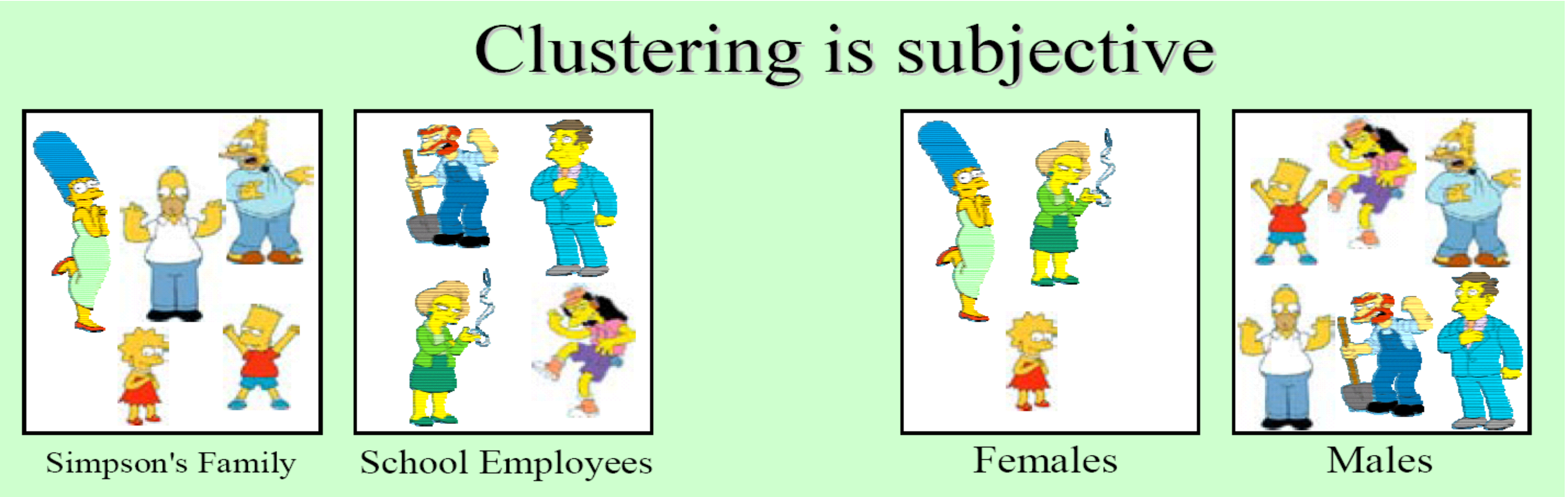
Clustering:

The process of grouping a set of objects into classes of similar objects

–high intra-class similarity

–low inter-class similarity

–It is the commonest form of unsupervised learning



What is Similarity?

Hard to define! *But we know it when we see it*

The real meaning of similarity is a philosophical question. We will take a more pragmatic approach: think in terms of a **distance** (rather than similarity) between random variables.

The K- means Clustering Problem

K-means Clustering Problem

Given a set of observations $(x_1, x_2, \ldots, x_n)$, where $x_i \in \mathbb{R}^d$

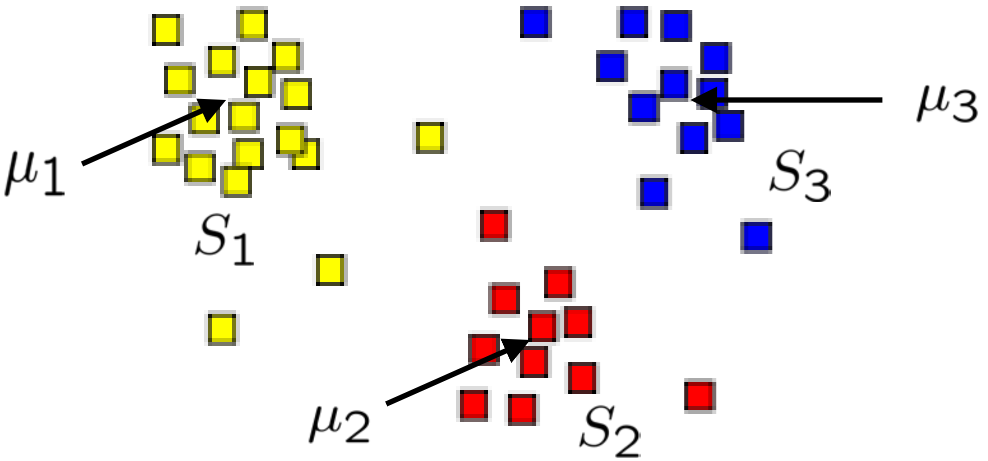
***K*-means clustering problem:**

Partition the *n* observations into *K* sets ($K \le n$) $\mathbf{S} = \{S_1, S_2, \ldots, S_K\}$ such that the sets minimize the within-cluster sum of squares:

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{K} \sum_{\mathbf{x}_j \in S_i} \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2$$

where μ_i is the mean of points in set S_i.

K=3



K-means Clustering Problem

Given a set of observations $(x_1, x_2, \ldots, x_n)$, where $x_i \in \mathbb{R}^d$

***K*-means clustering problem:**

Partition the *n* observations into *K* sets ($K \leq n$) $\mathbf{S} = \{S_1, S_2, \ldots, S_K\}$ such that the sets minimize the within-cluster sum of squares:

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{K} \sum_{\mathbf{x}_j \in S_i} \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2$$

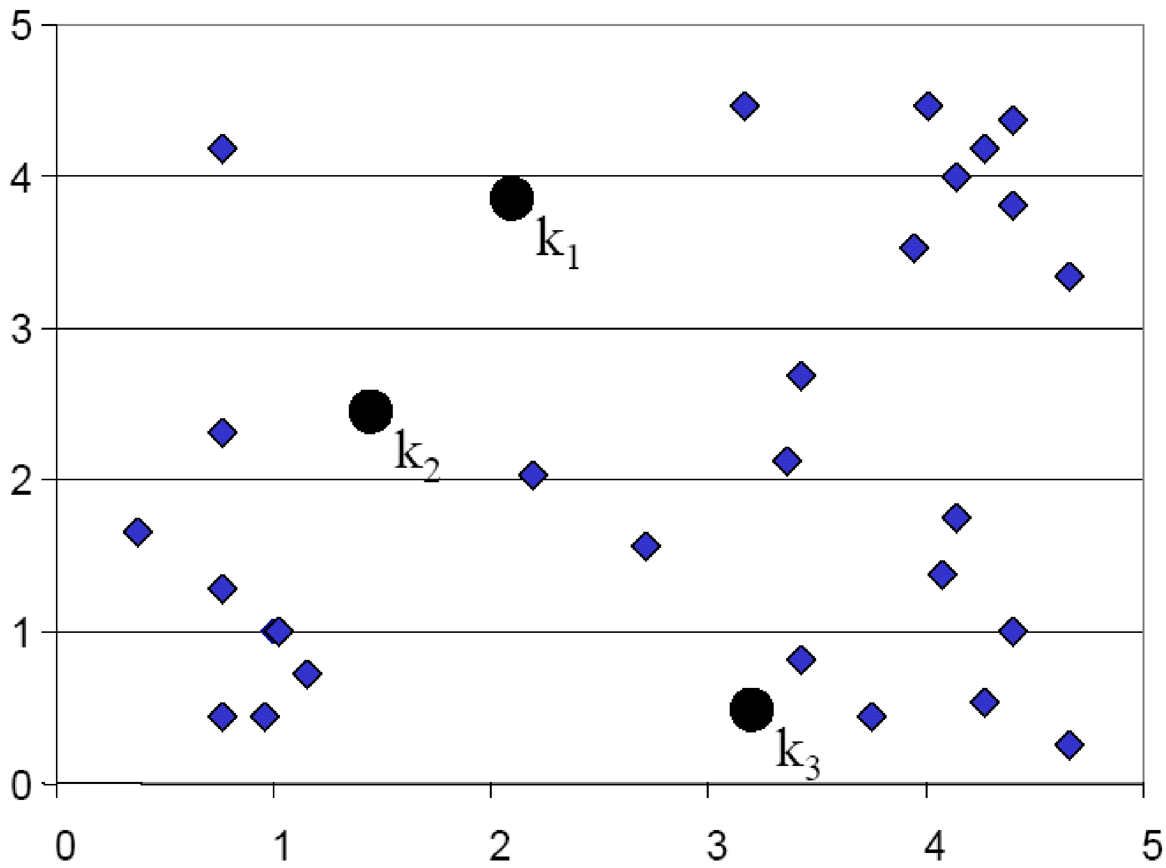
where μ_i is the mean of points in set S_i.

How hard is this problem?

The problem is NP hard, but there are good heuristic algorithms that seem to work well in practice:

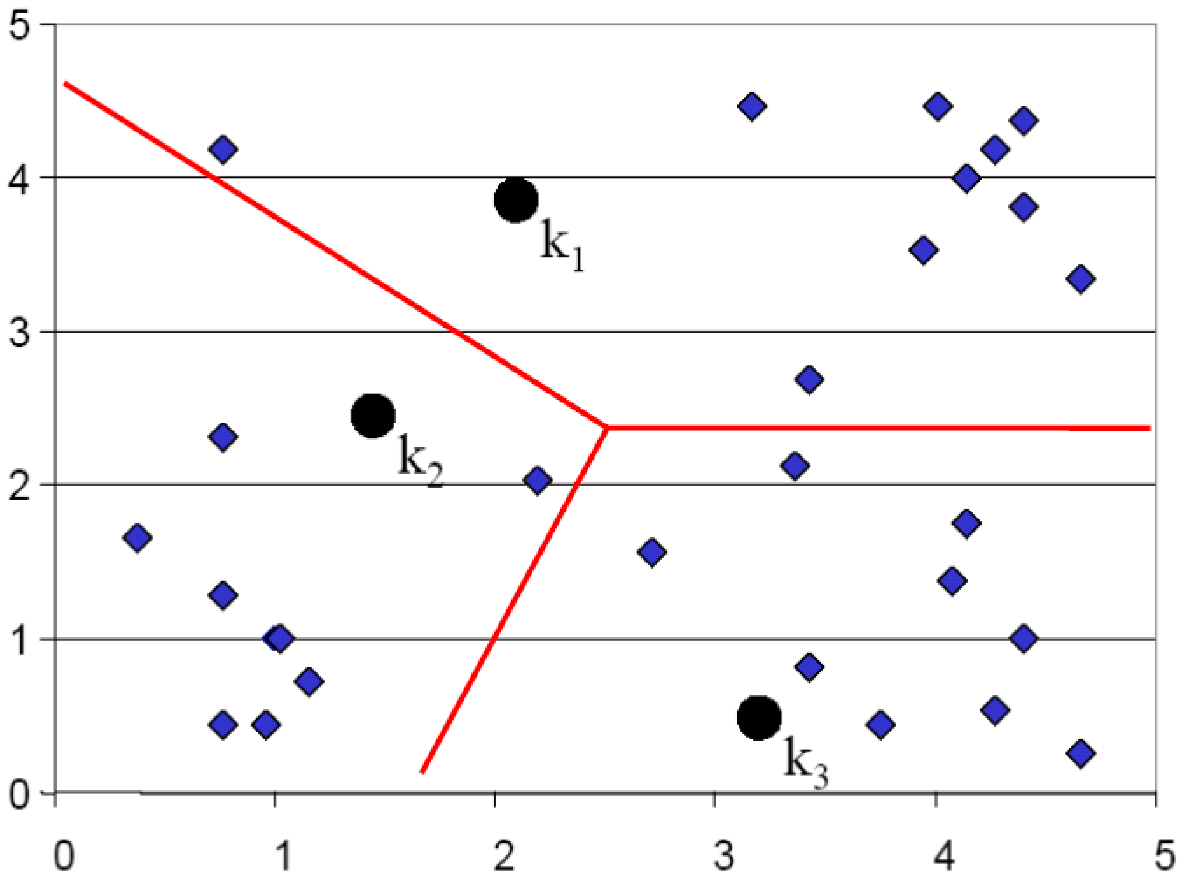
- K–means algorithm
- mixture of Gaussians

K-means Clustering Alg: Step 1



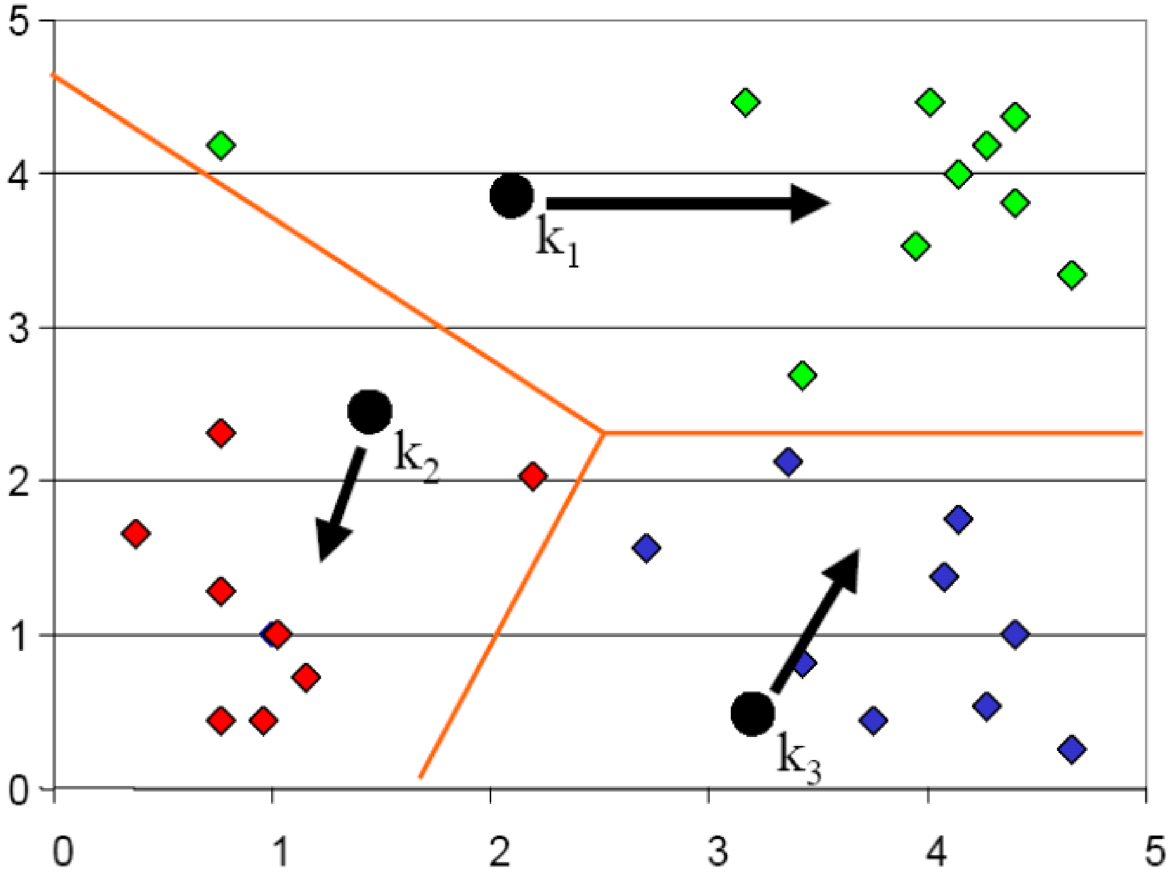
- Given n objects.
- Guess the cluster centers k_1, k_2, k_3. (They were $\mu_1,\ldots,\mu_3$ in the previous slide)

K-means Clustering Alg: Step 2



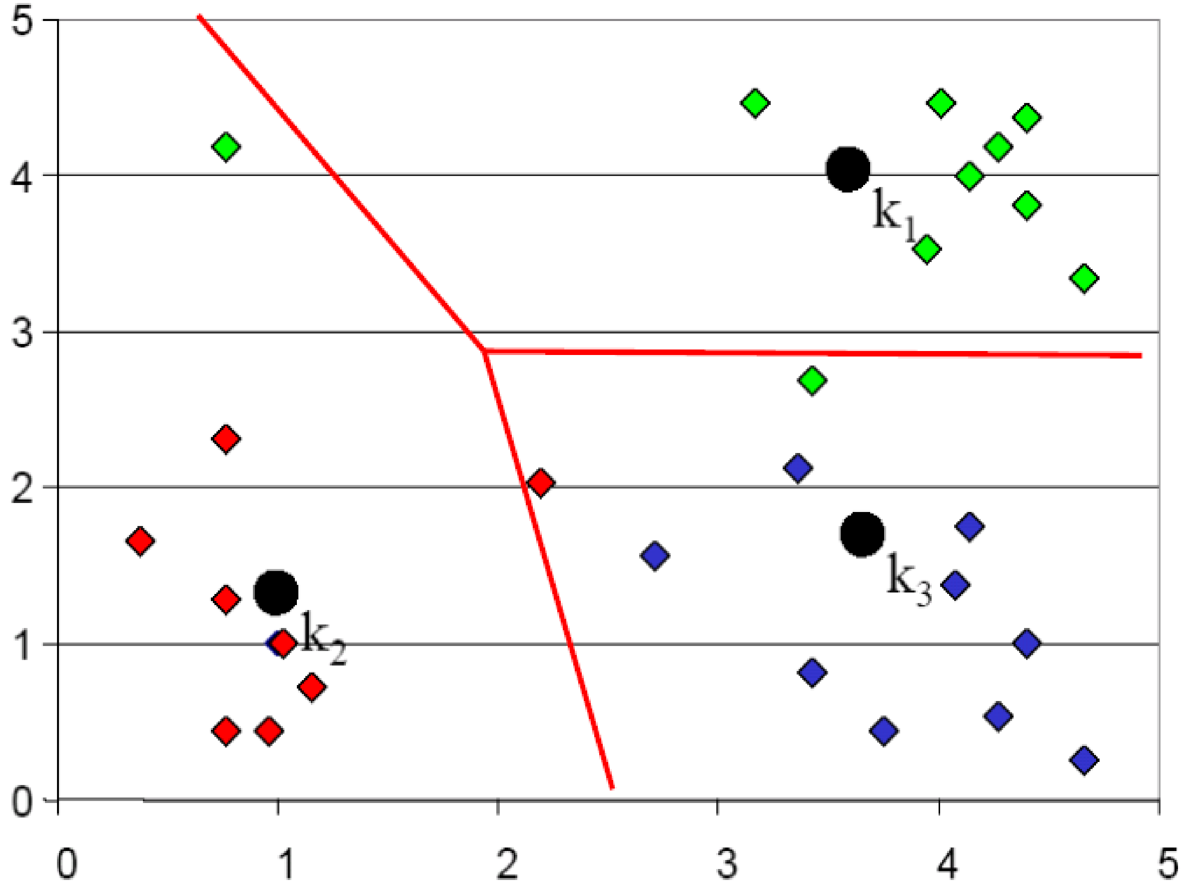
- Build a Voronoi diagram based on the cluster centers k_1, k_2, k_3.
- Decide the class memberships of the n objects by assigning them to the nearest cluster centers k_1, k_2, k_3.

K-means Clustering Alg: Step 3



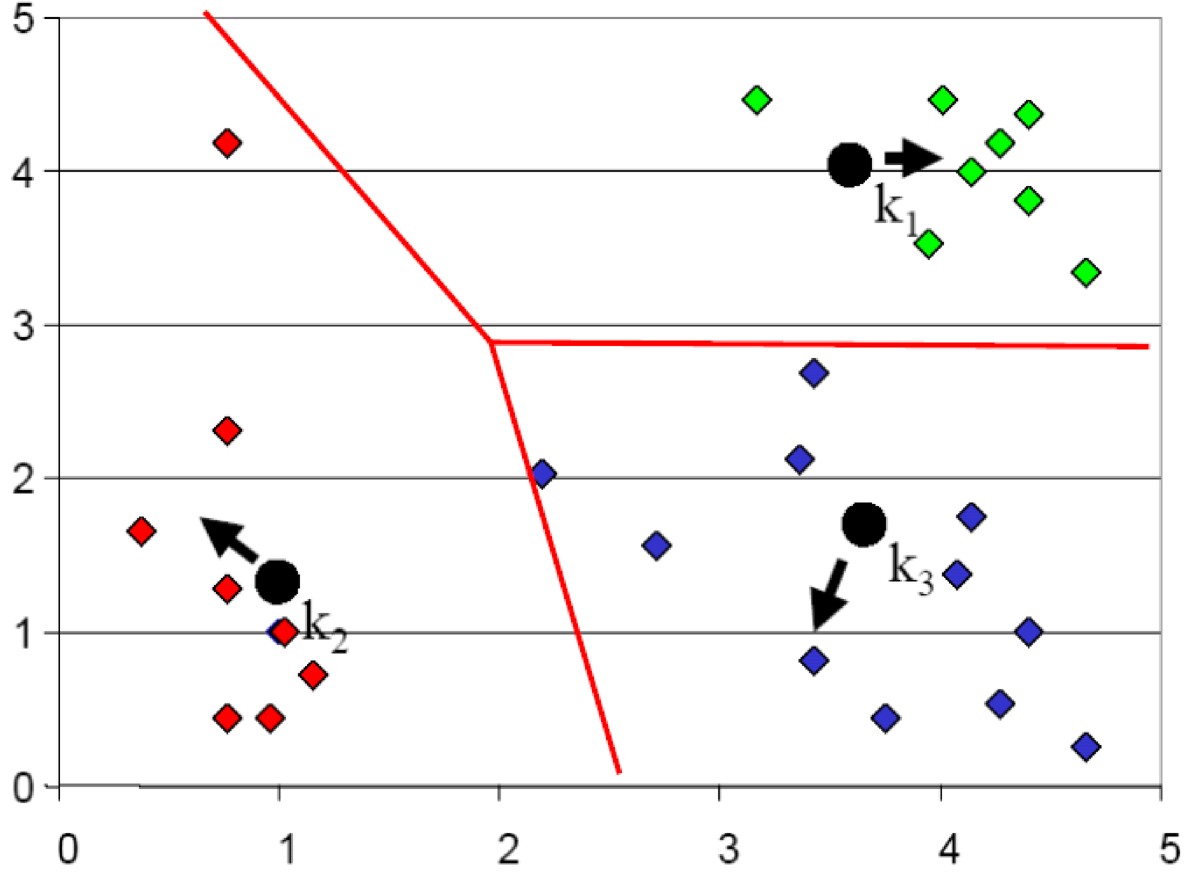
- Re-estimate the cluster centers (aka the centroid or mean), by assuming the memberships found above are correct.

K-means Clustering Alg: Step 4



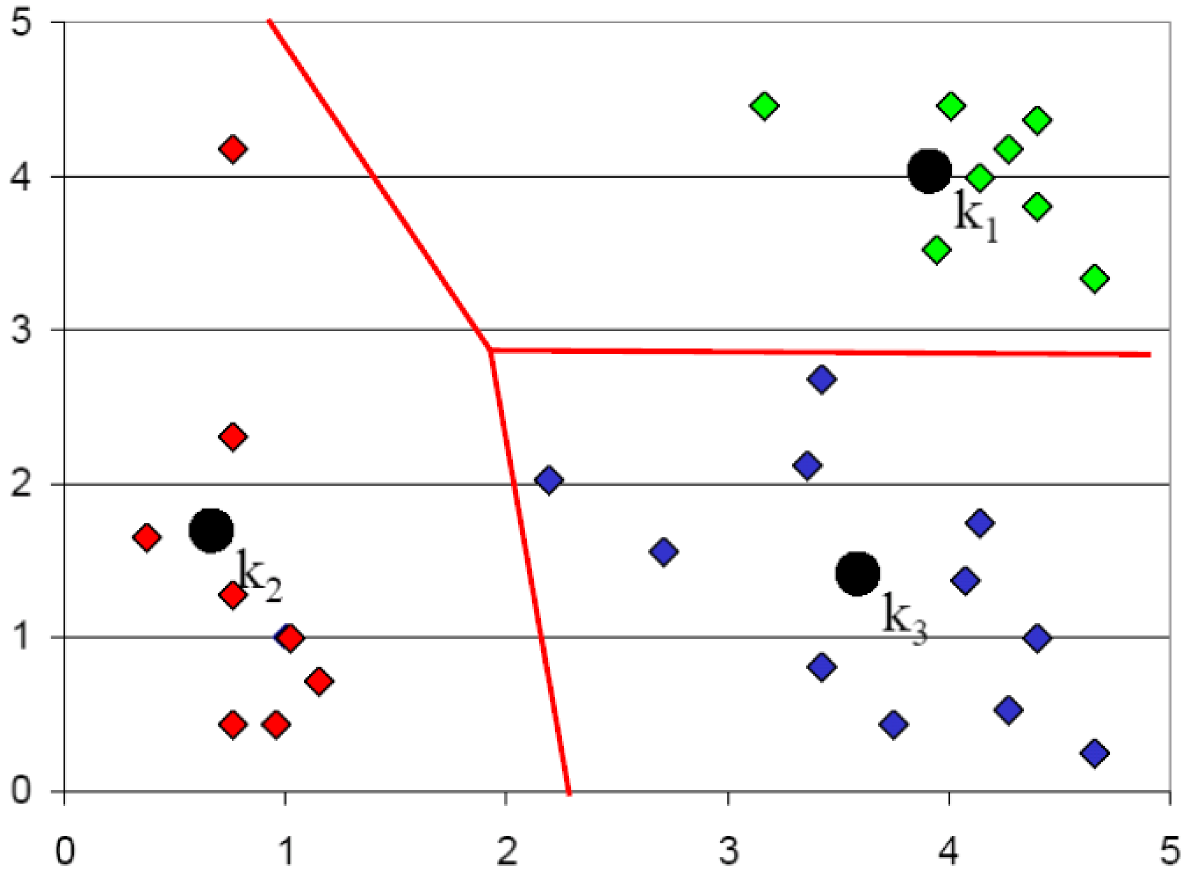
- Build a new Voronoi diagram.
- Decide the class memberships of the n objects based on this diagram

K-means Clustering Alg: Step 5



- Re-estimate the cluster centers.

K-means Clustering Alg: Step 6



- Stop when everything is settled.
  (The Voronoi diagrams don't change anymore)

K- means Clustering Algorithm

Algorithm

Input

– Data + Desired number of clusters, K

Initialize

– the K cluster centers (randomly if necessary)

Iterate

1. Decide the class memberships of the n objects by assigning them to the nearest cluster centers

2. Re-estimate the K cluster centers (aka the centroid or mean), by assuming the memberships found above are correct.

Termination

– If none of the n objects changed membership in the last iteration, exit.

Otherwise go to 1.

K- means Algorithm Computation Complexity

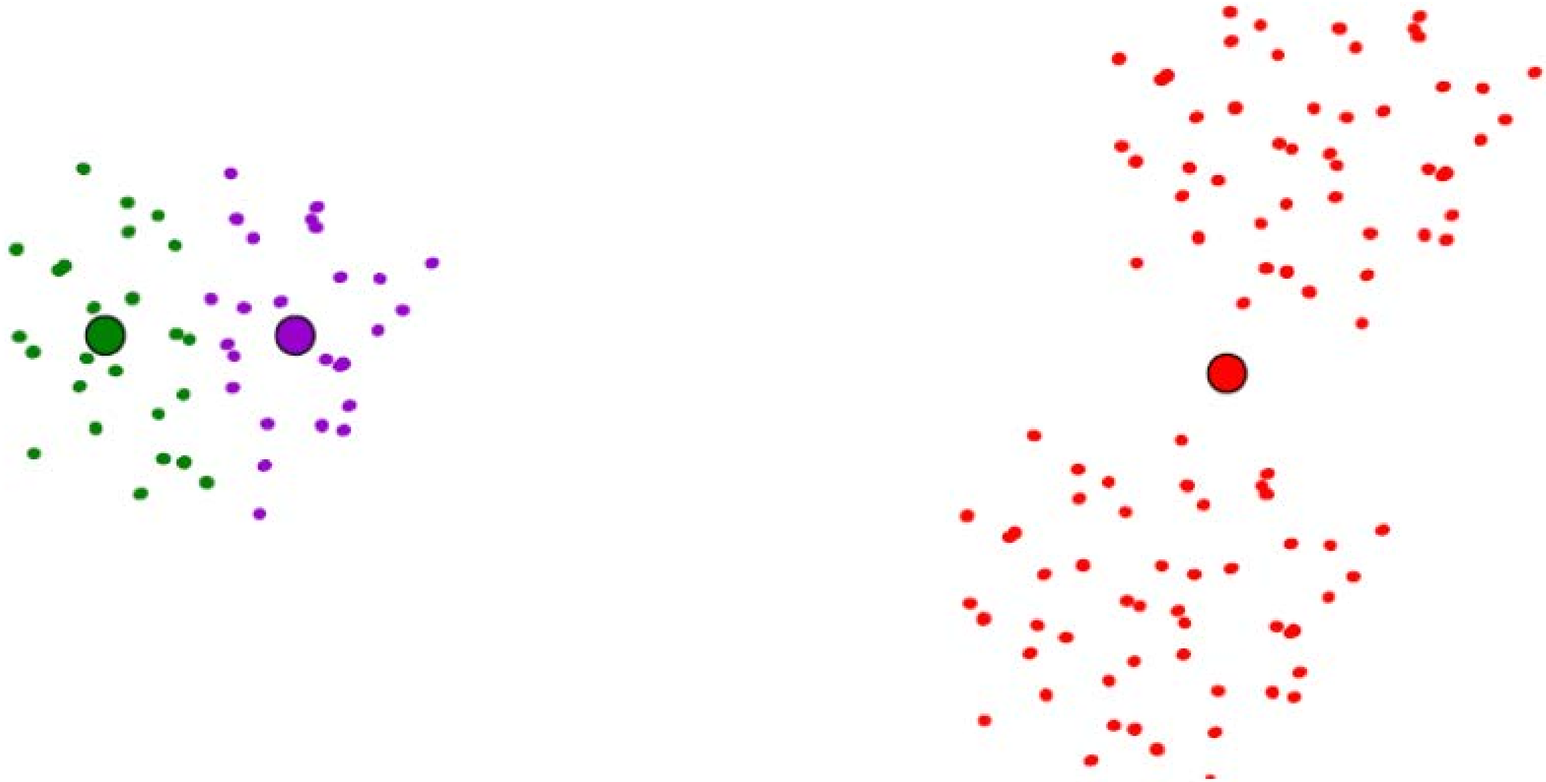
- At each iteration,
  - – Computing distance between each of the n objects and the K cluster centers is $O(Kn)$.
  - – Computing cluster centers: Each object gets added once to some cluster: $O(n)$.

- Assume these two steps are each done once for ℓ iterations: $O(\ell Kn)$.

Can you prove that the K-means algorithm guaranteed to terminate?

Seed Choice

Seed Choice



Seed Choice

The results of the K- means Algorithm can vary based on random seed selection.

- Some seeds can result in **poor convergence rate**, or convergence to **sub-optimal** clustering.
- K-means algorithm can get stuck easily in **local minima.**
  – Select good seeds using a heuristic (e.g., object least similar to any existing mean)
  – Try out **multiple** starting points (very important!!!)
  – Initialize with the results of another method.

Alternating Optimization

K- means Algorithm (more formally)

- **Randomly initialize k centers**

$$\mu^0 = (\mu_1^0, \ldots, \mu_K^0)$$

- **Classify**: At iteration t, assign each point j $\in$ {1,...,n} to nearest center:

$$C^t(j) \leftarrow \arg\min_i \|\mu_i^t - x_j\|^2$$

Classification at iteration *t*

- **Recenter**: μ_i is the centroid of the new sets:

$$\mu_i^{(t+1)} \leftarrow \arg\min_{\mu} \sum_{j:C^t(j)=i} \|\mu - x_j\|^2$$

Re-assign new cluster centers at iteration *t*

What is K-means optimizing?

- Define the following potential function F of centers μ and point allocation C

$$\mu = (\mu_1, \ldots, \mu_K)$$

$$C = (C(1), \ldots, C(n))$$

$$\left.\begin{aligned} F(\mu, C) &= \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2 \\ &= \sum_{i=1}^{K} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2 \end{aligned}\right\} \text{Two equivalent versions}$$

- Optimal solution of the K-means problem:

$$\boxed{\min_{\mu, C} F(\mu, C)}$$

K-means Algorithm

Optimize the potential function:

$$\min_{\mu,C} F(\mu, C) = \min_{\mu,C} \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2 = \min_{\mu,C} \sum_{i=1}^{K} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2$$

K-means algorithm:

(1) Fix μ, Optimize C

$$\min_{C(1),C(2),\ldots,C(n)} \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2 = \sum_{j=1}^{n} \underbrace{\min_{C(j)} \|\mu_{C(j)} - x_j\|^2}_{\text{Exactly first step}}$$

Assign each point to the nearest cluster center

(2) Fix C, Optimize μ

$$\min_{\mu_1,\ldots,\mu_K} \sum_{i=1}^{K} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2 = \sum_{i=1}^{K} \underbrace{\min_{\mu_i} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2}_{\text{Exactly 2nd step (re-center)}}$$

K-means Algorithm

Optimize the potential function:

$$\min_{\mu,C} F(\mu, C) = \min_{\mu,C} \sum_{j=1}^{n} \|\mu_{C(j)} - x_j\|^2$$

K-means algorithm: (coordinate descent on F)

(1) Fix μ, Optimize C **Expectation step**

(2) Fix C, Optimize μ **Maximization step**

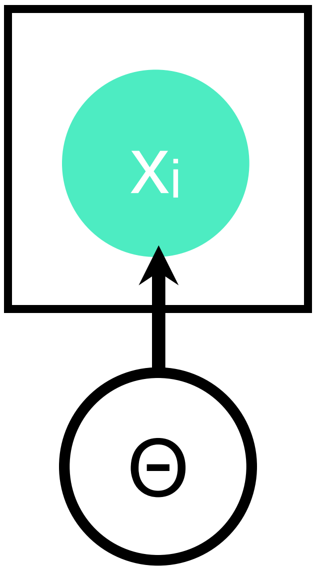
Today, we will see a generalization of this approach:

EM algorithm

Gaussian Mixture Model

Density Estimation

Generative approach



$$p(x_1, \ldots, x_n|\theta) = \prod_{i=1}^{n} p(x_i|\theta)$$

- There is a latent parameter Θ
- For all i, draw observed x_i given Θ

What if the basic model doesn't fit all data?

⇒ Mixture modelling, Partitioning algorithms

Different parameters for different parts of the domain. $[\theta_1, \ldots, \theta_K]$

Partitioning Algorithms

- **K-means**
  - –**hard assignment**: each object belongs to only one cluster

$$\theta_i \in \{\theta_1, \ldots, \theta_K\}$$

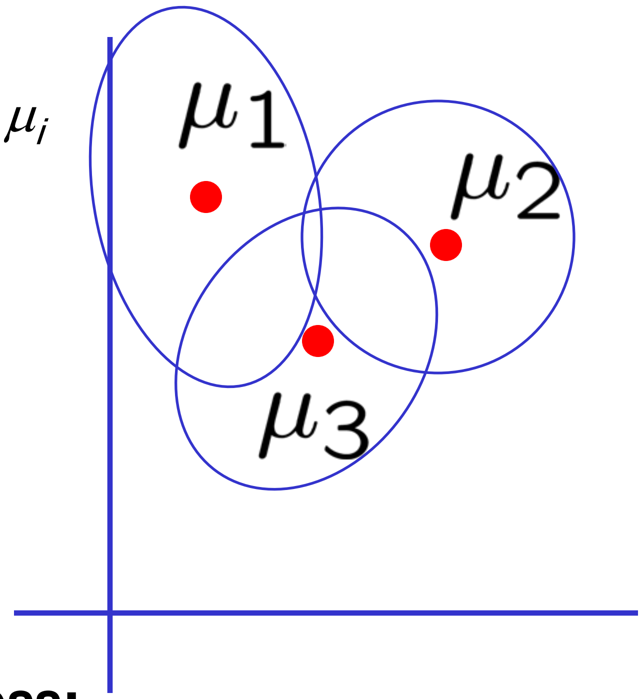
- **Mixture modeling**
  - –**soft assignment**: probability that an object belongs to a cluster

$$(\pi_1, \ldots, \pi_K),\ \pi_i \geq 0,\ \sum_{i=1}^{K} \pi_i = 1$$

Gaussian Mixture Model

Mixture of K Gaussians distributions: (Multi-modal distribution)

- There are K components
- Component *i* has an associated mean vector μ_i



Component *i* generates data from $N(\mu_i, \Sigma_i)$

Each data point is generated using this process:

1) Choose component i with probability $\pi_i = P(y = i)$
2) Datapoint $x \sim N(\mu_i, \Sigma_i)$

Gaussian Mixture Model

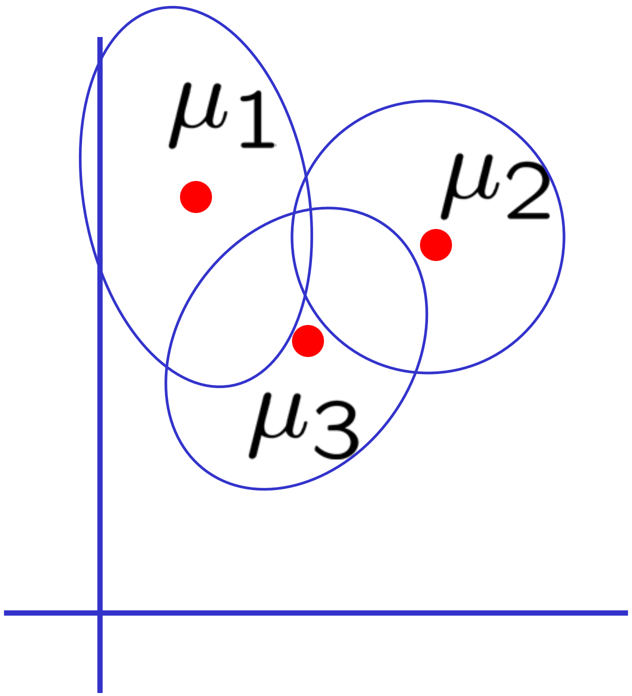
Mixture of K Gaussians distributions: (Multi-modal distribution)

Hidden variable

$$p(x|y=i) = N(\mu_i, \Sigma_i)$$

$$p(x) = \sum_{i=1}^{K} p(x|y=i)P(y=i)$$

Observed data **Mixture component** **Mixture proportion**



Mixture of Gaussians Clustering

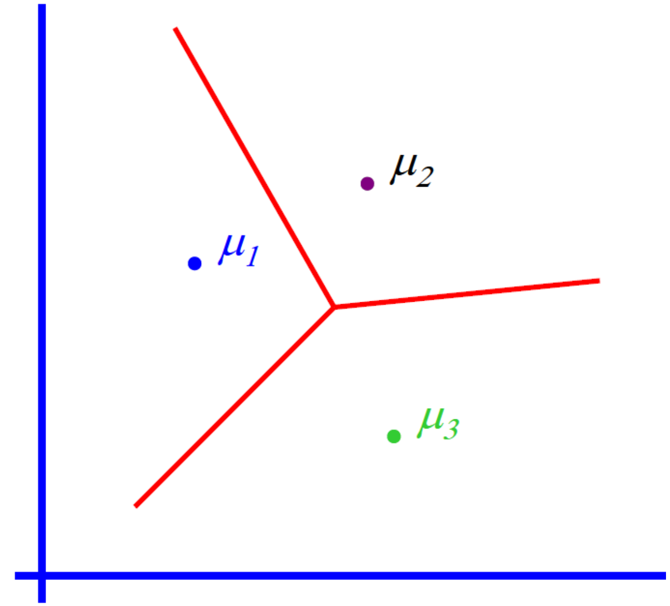
Assume that

$\Sigma_i = \sigma^2 \mathbf{I}$, for simplicity.

$p(x|y=i) = N(\mu_i, \sigma^2 \mathbf{I})$

$p(y=i) = \pi_i$

$\mu_1, \ldots, \mu_K, \sigma^2, \pi_1, \ldots, \pi_K$ are known.



Cluster x based on posteriors:

$$\log \frac{P(y=i|x)}{P(y=j|x)}$$

$$= \log \frac{p(x|y=i)P(y=i)/p(x)}{p(x|y=j)P(y=j)/p(x)}$$

$$= \log \frac{p(x|y=i)\pi_i}{p(x|y=j)\pi_j} = \log \frac{\pi_i \exp(\frac{-1}{2\sigma^2}\|x-\mu_i\|^2)}{\pi_j \exp(\frac{-1}{2\sigma^2}\|x-\mu_j\|^2)} = w^T x$$

$\boxed{\text{Depends on } \mu_1, \ldots, \mu_k, \sigma^2, \pi_1, \ldots, \pi_k}$

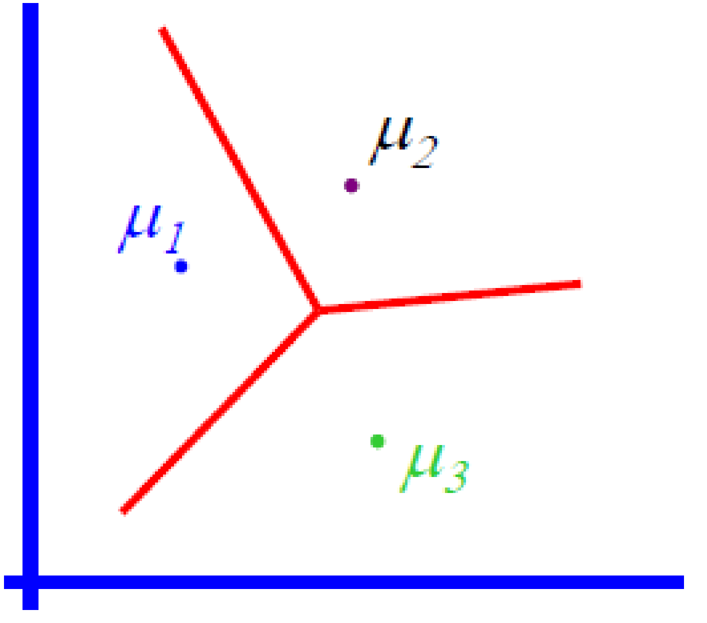
"Linear Decision boundary" – Since the second-order terms cancel out

MLE for GMM

What if we don't know $\mu_1, \ldots, \mu_K, \sigma^2, \pi_1, \ldots, \pi_K$?

$\Rightarrow$ **Maximum Likelihood Estimate (MLE)**

$$\theta = [\mu_1, \ldots, \mu_K, \sigma^2, \pi_1, \ldots, \pi_K]$$



$$\arg\max_\theta \prod_{j=1}^{n} P(x_j|\theta)$$

$$= \arg\max_\theta \prod_{j=1}^{n} \sum_{i=1}^{K} P(y_j = i, x_j|\theta)$$

$$= \arg\max_\theta \prod_{j=1}^{n} \sum_{i=1}^{K} P(y_j = i|\theta) p(x_j|y_j = i|\theta)$$

$$= \arg\max_\theta \prod_{j=1}^{n} \sum_{i=1}^{K} \pi_i \frac{1}{\sqrt{2\pi\sigma^2}} \exp(\frac{-1}{2\sigma^2}\|x_j - \mu_i\|^2)$$

K-means and GMM

- Assume data comes from a mixture of K Gaussians distributions with **same variance** σ^2
- Assume **Hard assignment**:

$$P(y_j = i) = 1 \text{ if } i = C(j)$$
$$= 0 \text{ otherwise}$$

Maximize marginal likelihood (MLE):

$$\arg\max_\theta \prod_{j=1}^{n} P(x_j|\theta) = \arg\max_\theta \prod_{j=1}^{n} \sum_{i=1}^{K} \overbrace{P(y_j = i) \frac{1}{\sqrt{2\pi\sigma^2}} \exp(\frac{-1}{2\sigma^2}\|x_j - \mu_i\|^2)}^{P(y_j = i, x_j|\theta)}$$

$$= \arg\max_\theta \prod_{j=1}^{n} \exp(\frac{-1}{2\sigma^2}\|x_j - \mu_{C(j)}\|^2)$$

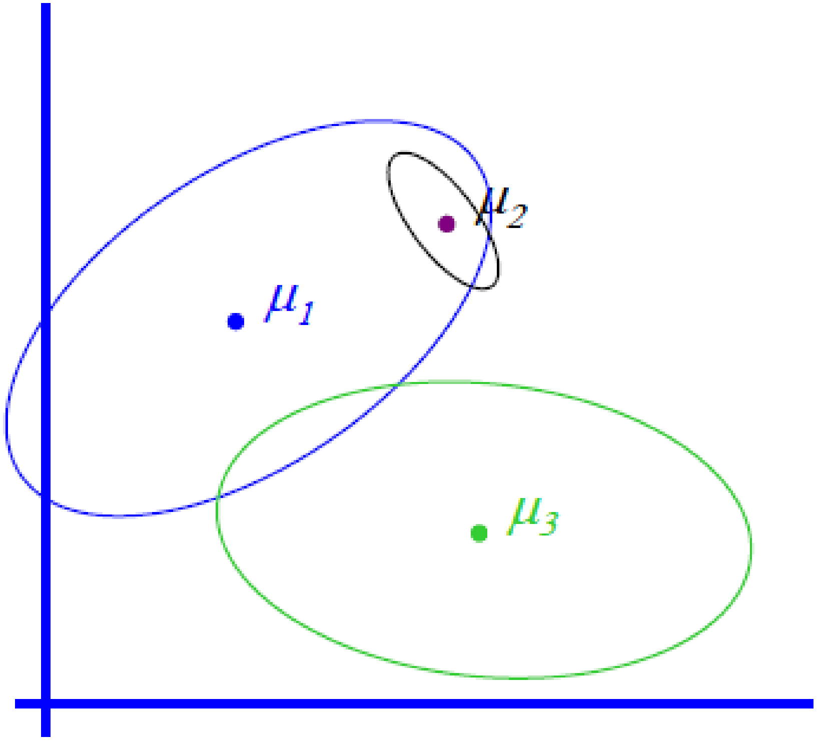
$$= \arg\min_{\mu,C} \sum_{j=1}^{n} \|x_j - \mu_{C(j)}\|^2) = \arg\min_{\mu,C} F(\mu, C)$$

Same as K-means!!!

General GMM

General GMM –Gaussian Mixture Model (Multi-modal distribution)

- There are k components
- Component *i* has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix Σ_i. Each data point is generated according to the following recipe:



1) Pick a component at random: Choose component i with probability *P(y=i)*

2) Datapoint $x \sim N(\mu_i, \Sigma_i)$

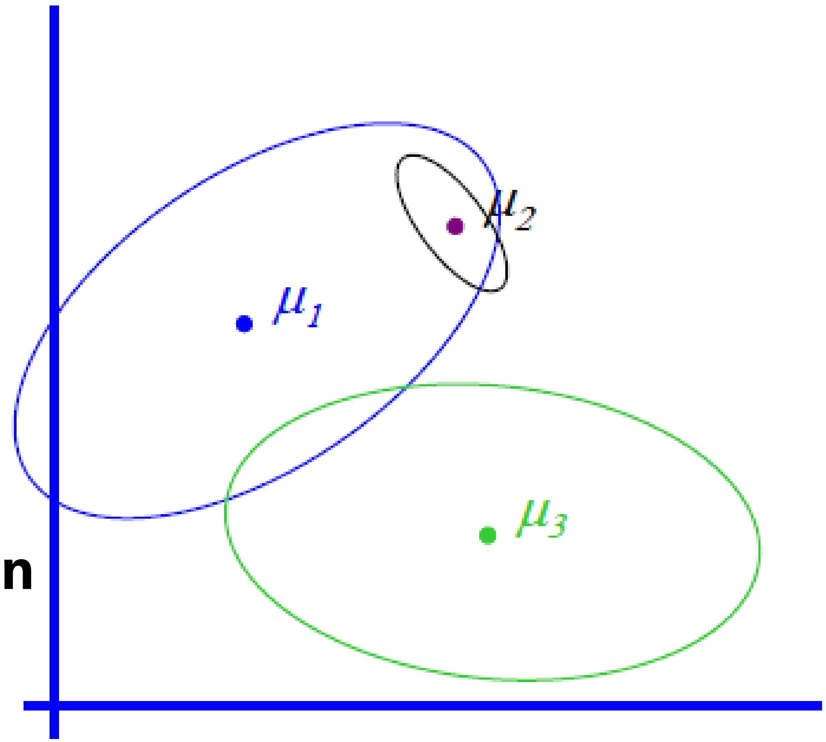
General GMM

GMM –Gaussian Mixture Model (Multi-modal distribution)

$$p(x|y=i) = N(\mu_i, \Sigma_i)$$

$$p(x) = \sum_{i=1}^{K} p(x|y=i)P(y=i)$$

Mixture component — **Mixture proportion**



General GMM

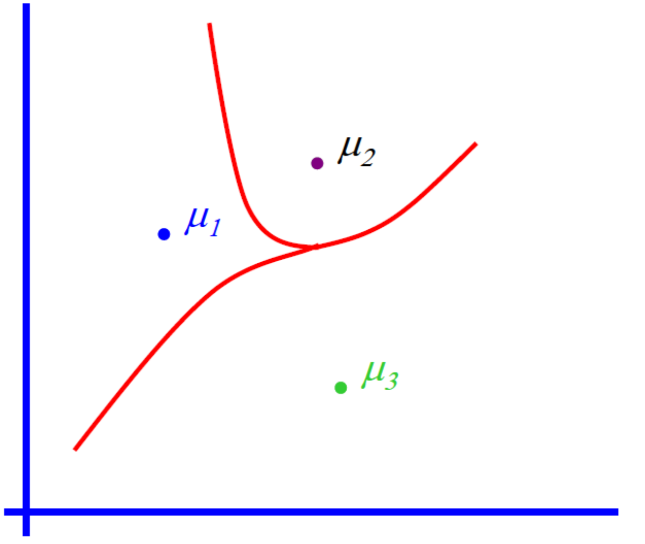
Assume that

$\theta = [\mu_1, \ldots, \mu_K, \Sigma_1, \ldots, \Sigma_K, \pi_1, \ldots, \pi_K]$ are known.

$$p(x|y=i) = N(\mu_i, \Sigma_i)$$

$$p(y=i) = \pi_i$$

Clustering based on posteriors:



$$\log \frac{P(y=i|x)}{P(y=j|x)}$$

$$= \log \frac{p(x|y=i)P(y=i)/p(x)}{p(x|y=j)P(y=j)/p(x)}$$

$$= \log \frac{p(x|y=i)\pi_i}{p(x|y=j)\pi_j} = \log \frac{\pi_i \frac{1}{\sqrt{2\pi|\Sigma_i|}} \exp\left[-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)\right]}{\pi_j \frac{1}{\sqrt{2\pi|\Sigma_j|}} \exp\left[-\frac{1}{2}(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j)\right]}$$

$$= x^T W x + w^T x + c$$

Depends on $\mu_1, \ldots, \mu_K, \Sigma_1, \ldots, \Sigma_K, \pi_1, \ldots, \pi_K$

“Quadratic Decision boundary” – second-order terms don’t cancel out

General GMM MLE Estimation

What if we don't know $\theta = [\mu_1, \ldots, \mu_K, \Sigma_1, \ldots, \Sigma_K, \pi_1, \ldots, \pi_K]$?

$\Rightarrow$ **Maximize marginal likelihood (MLE):**

$$\arg\max_\theta \prod_{j=1}^n P(x_j|\theta) = \arg\max_\theta \prod_{j=1}^n \sum_{i=1}^K P(y_j = i, x_j|\theta)$$

$$= \arg\max_\theta \prod_{j=1}^n \sum_{i=1}^K P(y_j = i|\theta) p(x_j|y_j = i|\theta)$$

$$= \arg\max_\theta \prod_{j=1}^n \sum_{i=1}^K \pi_i \frac{1}{\sqrt{2\pi|\Sigma_i|}} \exp\left[-\frac{1}{2}(x_j - \mu_i)^T \Sigma_i^{-1} (x_j - \mu_i)\right]$$

How do we find $\theta = [\mu_1, \ldots, \mu_K, \Sigma_1, \ldots, \Sigma_K, \pi_1, \ldots, \pi_K]$ which gives max. marginal likelihood?

* Set $\frac{\partial}{\partial \mu_i} \log \text{Prob}(\ldots) = 0$, and solve for μ_i. Non-linear, non-analytically solvable

* Use gradient descent. Doable, but often slow

* Use EM.

Expectation-Maximization (EM)

A general algorithm to deal with hidden data, but we will study it in the context of unsupervised learning (hidden class labels = clustering) first.

- EM is an optimization strategy for objective functions that can be interpreted as likelihoods in the presence of missing data.
- EM is much simpler than gradient methods:
  No need to choose step size.
- EM is an iterative algorithm with two linked steps:
  - E-step: fill-in hidden values using inference
  - M-step: apply standard MLE/MAP method to completed data
- We will prove that this procedure monotonically improves the likelihood (or leaves it unchanged). EM always converges to a local optimum of the likelihood.

Expectation-Maximization (EM)

A simple case:

- We have unlabeled data $x_1, x_2, \ldots, x_m$
- We know there are K classes
- We know $P(y=1)=\pi_1$, $P(y=2)=\pi_2$ $P(y=3)$ ... $P(y=K)=\pi_K$
- We know common variance σ^2
- We **don't** know $\mu_1, \mu_2, \ldots \mu_K$, and we want to learn them

We can write

$$p(x_1,\ldots,x_n|\mu_1,\ldots\mu_K) = \prod_{j=1}^{n} p(x_j|\mu_1,\ldots,\mu_K)$$ Independent data

$$= \prod_{j=1}^{n} \sum_{i=1}^{K} p(x_j, y_j = i|\mu_1,\ldots,\mu_K)$$ Marginalize over class

$$= \prod_{j=1}^{n} \sum_{i=1}^{K} p(x_j|y_j = i|\mu_1,\ldots,\mu_K)p(y_j = i)$$

$$\propto \prod_{j=1}^{n} \sum_{i=1}^{K} \exp(-\frac{1}{2\sigma^2}\|x_j - \mu_i\|^2)\pi_i$$ $\Rightarrow$ learn $\mu_1, \mu_2, \ldots \mu_K$

Expectation (E) step

We want to learn: $\theta = [\mu_1, \ldots, \mu_K]$

Our estimator at the end of iteration t-1: $\theta^{t-1} = [\mu_1^{t-1}, \ldots, \mu_K^{t-1}]$

At iteration t, construct function Q:

$$Q(\theta^t|\theta^{t-1}) = \sum_{j=1}^{n} \sum_{i=1}^{K} P(y_j = i|x_j, \theta^{t-1}) \log P(x_j, y_j = i|\theta^t)$$

E step

$$P(y_j = i|x_j, \theta^{t-1}) = P(y_j = i|x_j, \mu_1^{t-1}, \ldots, \mu_K^{t-1})$$

$$\propto P(x_j|y_j = i, \mu_1^{t-1}, \ldots, \mu_K^{t-1}) P(y_j = i)$$

$$\propto \exp(-\frac{1}{2\sigma^2}\|x_j - \mu_i^{t-1}\|^2)\pi_i$$

$$= \frac{\exp(-\frac{1}{2\sigma^2}\|x_j - \mu_i^{t-1}\|^2)\pi_i}{\sum_{i=1}^{K} \exp(-\frac{1}{2\sigma^2}\|x_j - \mu_i^{t-1}\|^2)\pi_i}$$

Equivalent to assigning clusters to each data point in K-means in a soft way

Maximization (M) step

$$Q(\theta^t|\theta^{t-1}) = \sum_{j=1}^{n}\sum_{i=1}^{K} P(y_j = i|x_j, \theta^{t-1}) \log P(x_j, y_j = i|\theta^t)$$

$$= \sum_{j=1}^{n}\sum_{i=1}^{K} P(y_j = i|x_j, \theta^{t-1})[\underbrace{\underbrace{\log P(x_j|y_j = i, \theta^t)}_{\propto \exp(-\frac{1}{2\sigma^2}\|x_j - \mu_i^t\|^2)} + \underbrace{\log P(y_j = i|\theta^t)}_{\pi_i}}_{\text{Joint distribution is simple}}]$$

We calculated these weights in the E step

$$R_{i,j}^{t-1} = P(y_j = i|x_j, \theta^{t-1})$$

Joint distribution is simple

M step At iteration t, maximize function Q in θ^t:

$$Q(\mu_i^t|\theta^{t-1}) \propto \sum_{j=1}^{n} R_{i,j}^{t-1}(-\frac{1}{2\sigma^2}\|x_j - \mu_i^t\|^2)$$

$$\frac{\partial}{\partial \mu_i^t} Q(\mu_i^t|\theta^{t-1}) = 0 \Rightarrow \sum_{j=1}^{n} R_{i,j}^{t-1}(x_n - \mu_i^t) = 0$$

$$\boxed{\mu_i^t = \sum_{j=1}^{n} w_j x_j \text{ where } w_j = \frac{R_{i,j}^{t-1}}{\sum_{j=1}^{n} R_{i,j}^{t-1}} = \frac{P(y_j=i|x_j,\theta^{t-1})}{\sum_{l=1}^{n} P(y_l=i|x_l,\theta^{t-1})}}$$

Equivalent to updating cluster centers in K-means

EM for spherical, same variance GMMs

E-step

Compute "expected" classes of all datapoints for each class

$$P(y_j = i | x_j, \theta^{t-1}) = \frac{\exp(-\frac{1}{2\sigma^2}\|x_j - \mu_i^{t-1}\|^2)\pi_i}{\sum_{i=1}^{K} \exp(-\frac{1}{2\sigma^2}\|x_j - \mu_i^{t-1}\|^2)\pi_i}$$

In K-means "E-step" we do hard assignment. EM does soft assignment

M-step

Compute Max. like **μ** given our data's class membership distributions (weights)

$$\mu_i^t = \sum_{j=1}^{n} w_j x_j \quad \text{where } w_j = \frac{P(y_j = i | x_j, \theta^{t-1})}{\sum_{l=1}^{n} P(y_l = i | x_l, \theta^{t-1})}$$

Iterate. Exactly the same as MLE with weighted data.

EM for general GMMs

The more general case:

- We have unlabeled data $x_1, x_2, \ldots, x_m$
- We know there are K classes
- We **don't** know $P(y=1)=\pi_1$, $P(y=2)=\pi_2$ $P(y=3)$ ... $P(y=K)=\pi_K$
- We **don't** know $\Sigma_1, \ldots \Sigma_K$
- We **don't** know $\mu_1, \mu_2, \ldots \mu_K$

We want to learn: $\theta = [\mu_1, \ldots, \mu_K, \pi_1, \ldots, \pi_K, \Sigma_1, \ldots, \Sigma_K]$

Our estimator at the end of iteration t-1:

$$\theta^{t-1} = [\mu_1^{t-1}, \ldots, \mu_K^{t-1}, \pi_1^{t-1}, \ldots, \pi_K^{t-1}, \Sigma_1^{t-1}, \ldots, \Sigma_K^{t-1}]$$

The idea is the same:

At iteration t, construct function Q (E step) and maximize it in θ^t (M step)

$$Q(\theta^t|\theta^{t-1}) = \sum_{j=1}^{n} \sum_{i=1}^{K} P(y_j = i|x_j, \theta^{t-1}) \log P(x_j, y_j = i|\theta^t)$$

EM for general GMMs

At iteration t, construct function Q (E step) and maximize it in θ^t (M step)

$$Q(\theta^t|\theta^{t-1}) = \sum_{j=1}^{n}\sum_{i=1}^{K} P(y_j = i|x_j, \theta^{t-1}) \log P(x_j, y_j = i|\theta^t)$$

E-step

Compute "expected" classes of all datapoints for each class

$$R_{i,j}^{t-1} = P(y_j = i|x_j, \theta^{t-1}) = \frac{\exp(-\frac{1}{2\sigma^2}\|x_j - \mu_i^{t-1}\|^2)\pi_i^{t-1}}{\sum_{i=1}^{K} \exp(-\frac{1}{2\sigma^2}\|x_j - \mu_i^{t-1}\|^2)\pi_i^{t-1}}$$

M-step $\frac{\partial}{\partial\theta^t} Q(\theta^t|\theta^{t-1}) = 0$

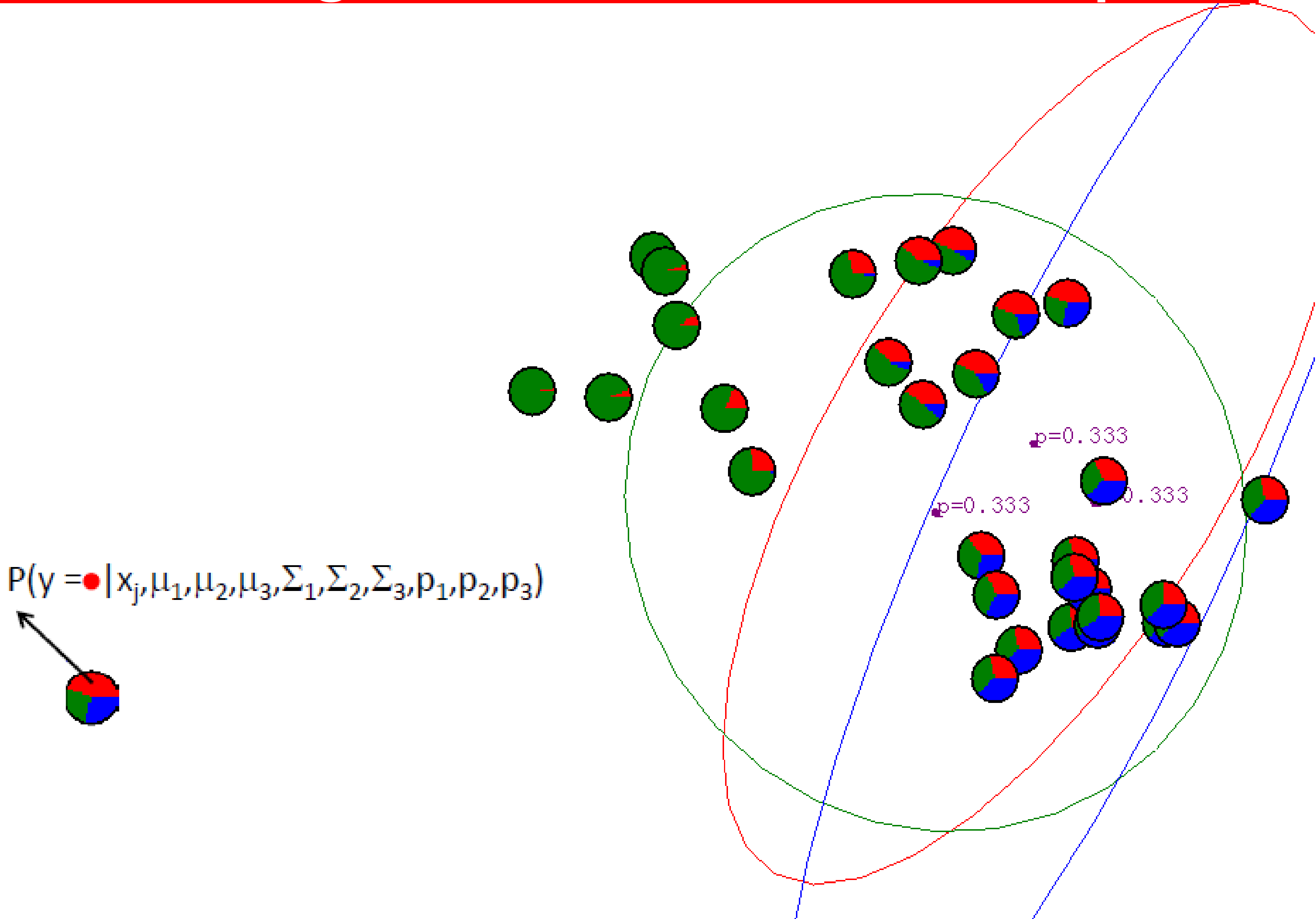
Compute MLEs given our data's class membership distributions (weights)

$$\mu_i^t = \sum_{j=1}^{n} w_j x_j \quad \text{where } w_j = \frac{R_{i,j}^{t-1}}{\sum_{j=1}^{n} R_{i,j}^{t-1}}$$

$$\Sigma_i^t = \sum_{j=1}^{n} w_j (x_j - \mu_i^t)^T (x_j - \mu_i^t)$$

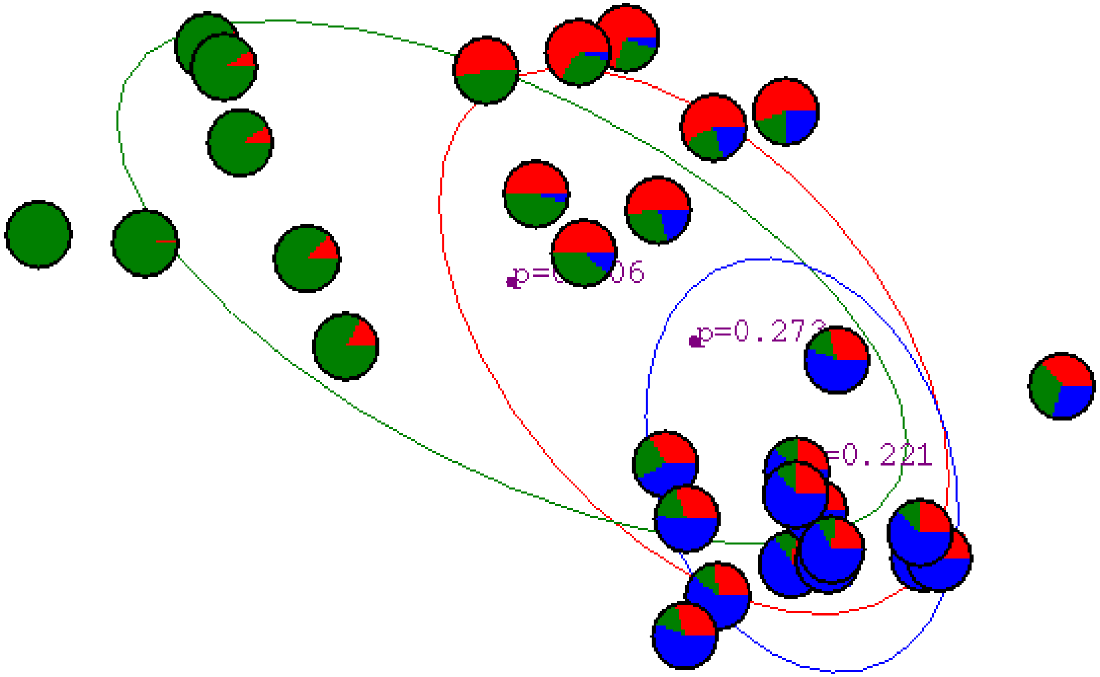
$$\pi_i^t = \frac{1}{n}\sum_{j=1}^{n} R_{i,j}^{t-1}$$

EM for general GMMs: Example



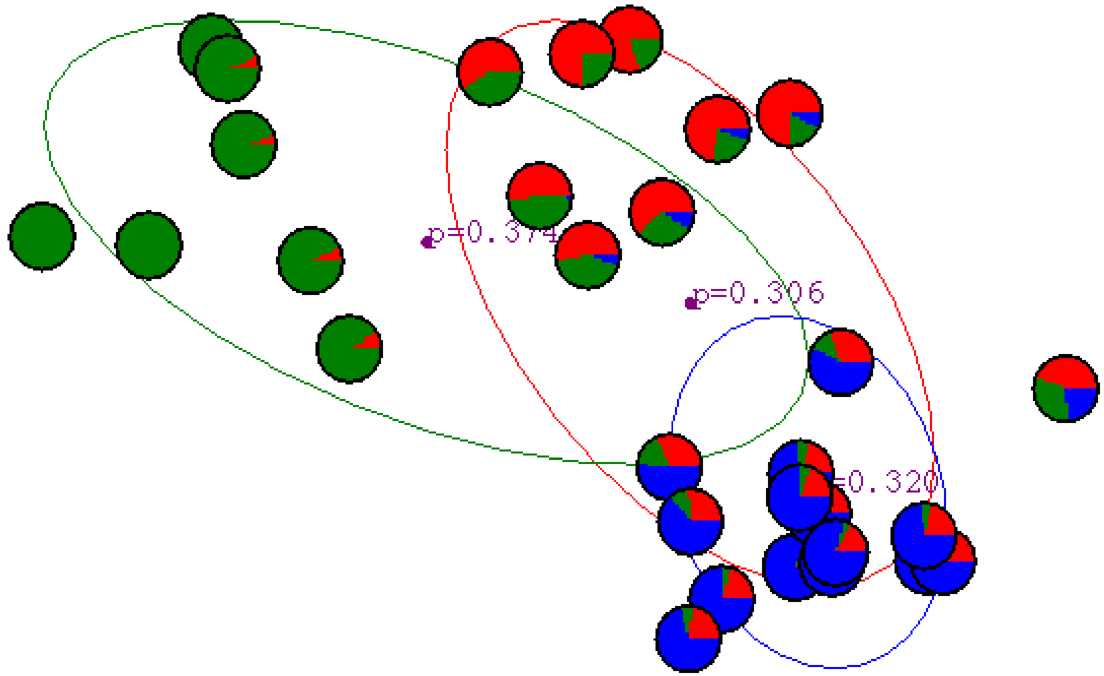
EM for general GMMs: Example

After 1st iteration



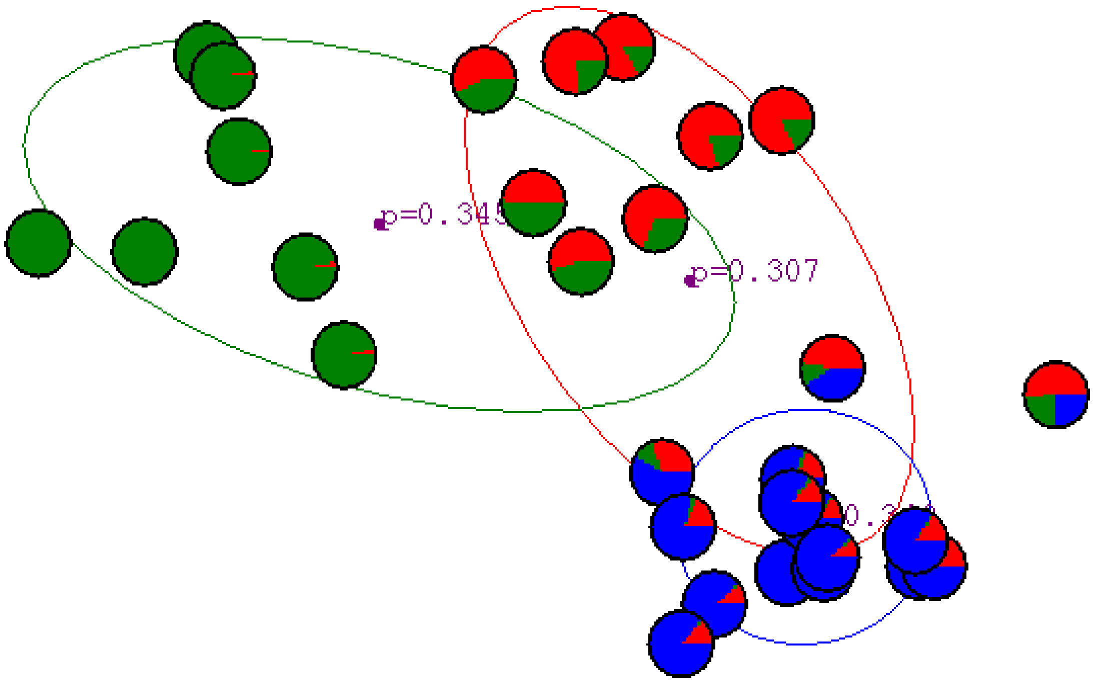
EM for general GMMs: Example

After 2nd iteration



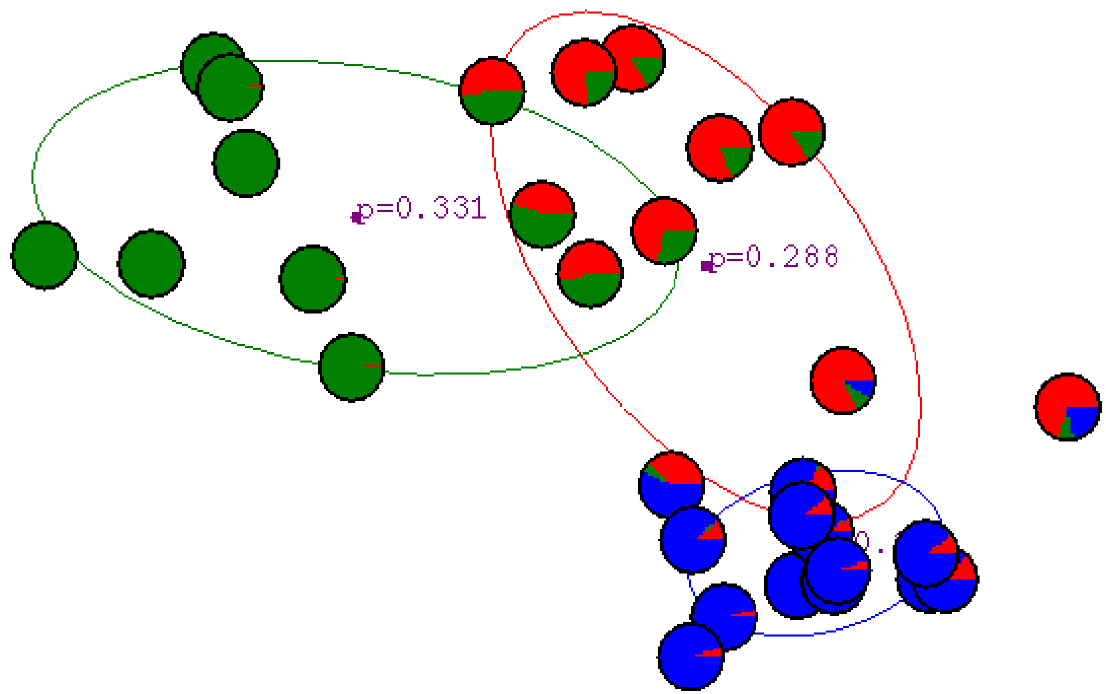
EM for general GMMs: Example

After 3rd iteration



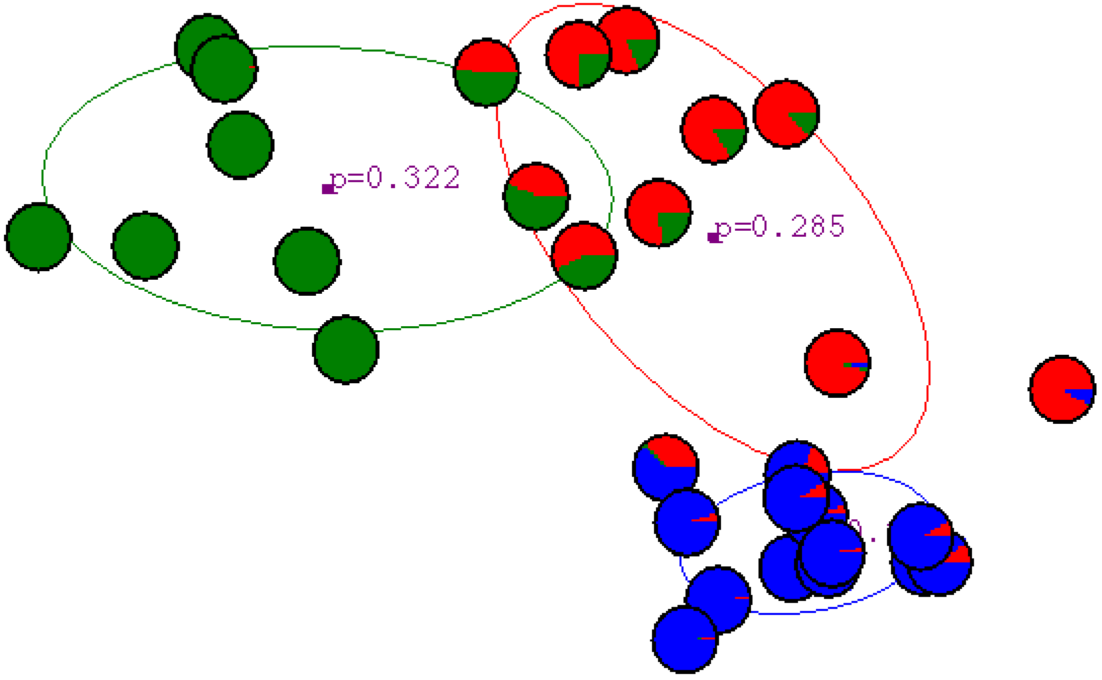
EM for general GMMs: Example

After 4th iteration



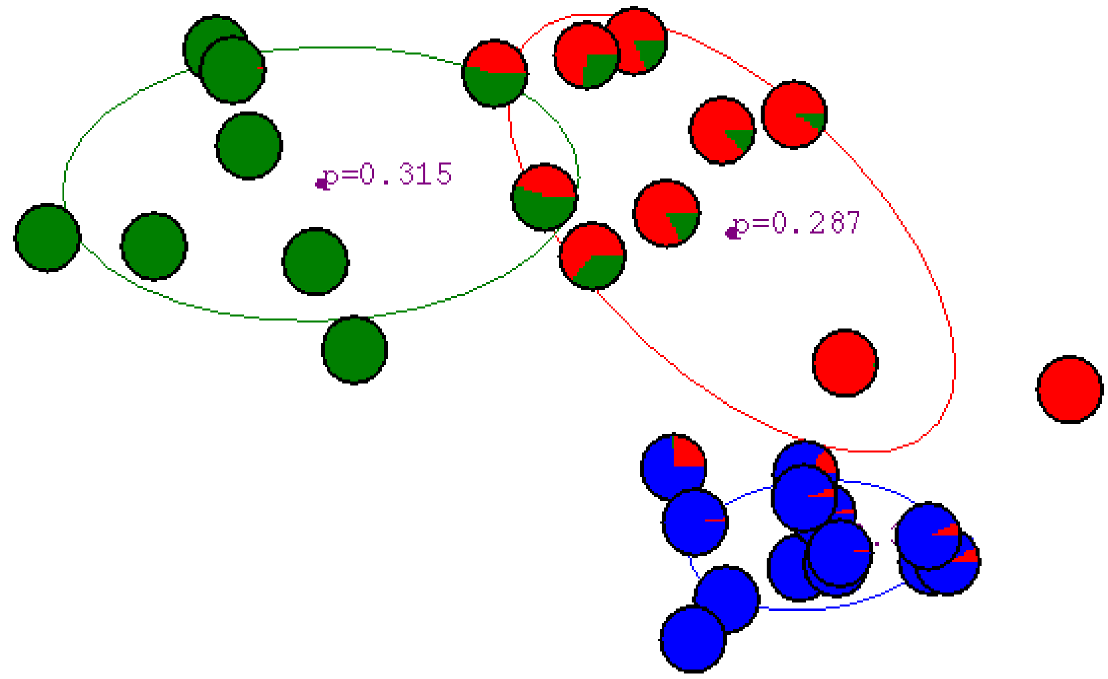
EM for general GMMs: Example

After 5th iteration



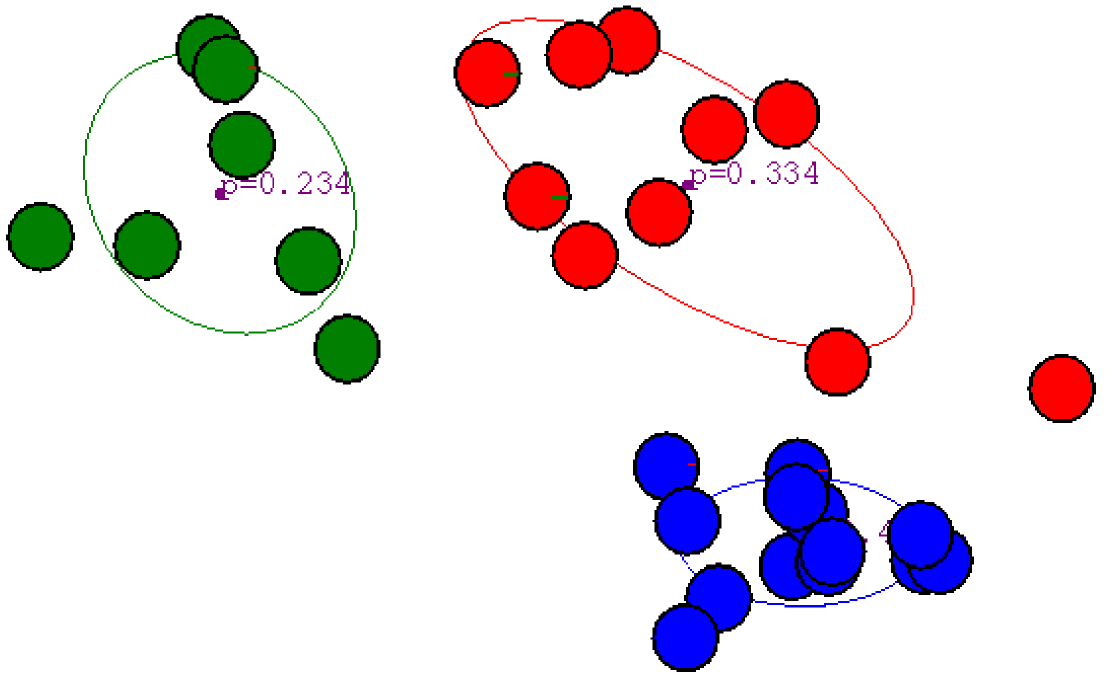
EM for general GMMs: Example

After 6th iteration

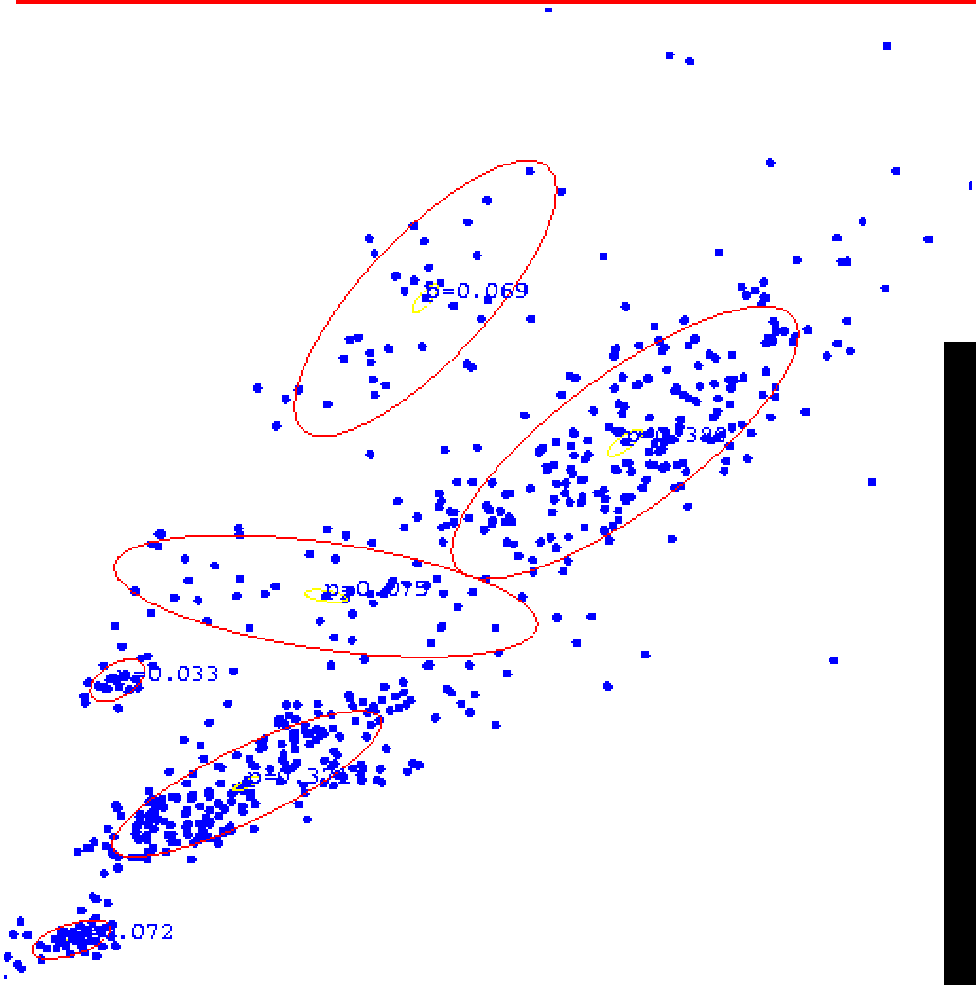


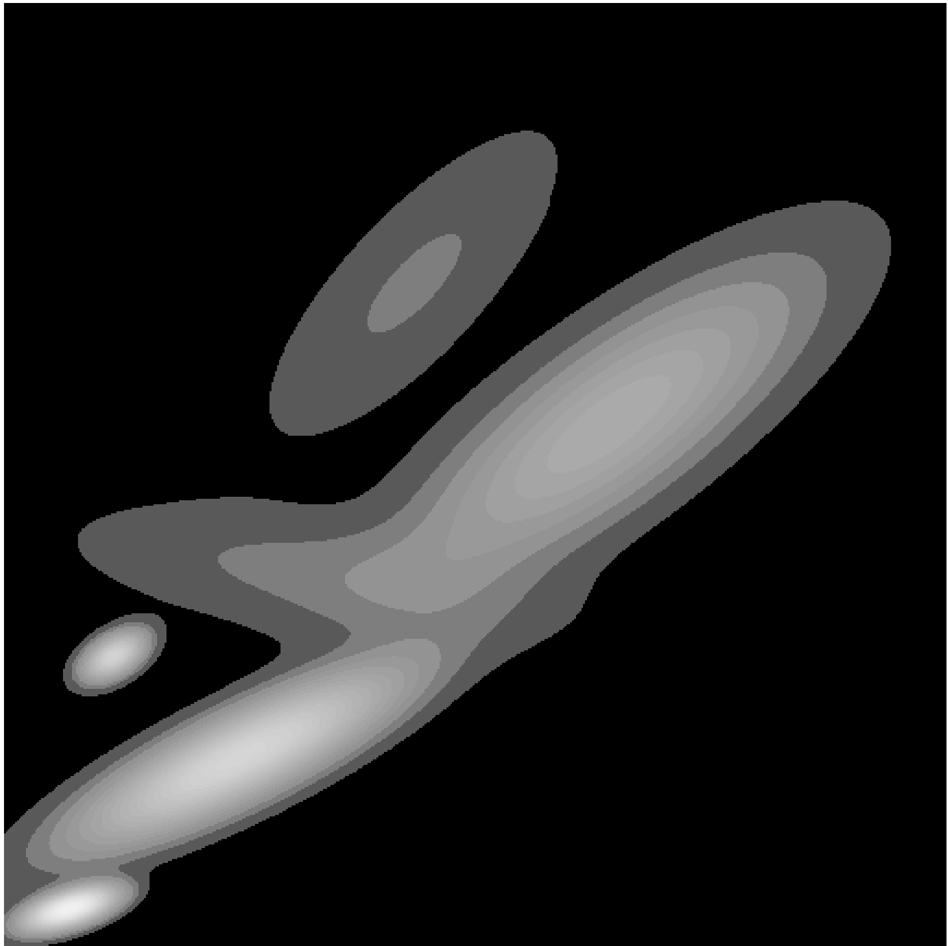
EM for general GMMs: Example

After 20th iteration



GMM for Density Estimation





General EM algorithm

What is EM in the general case, and why does it work?

General EM algorithm

Notation

Observed data: $D = \{x_1, \ldots, x_n\}$

Unknown variables: y

For example in clustering: $y = (y_1, \ldots, y_n)$

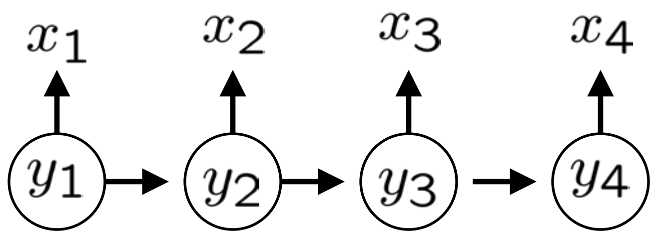
Paramaters: θ

For example in MoG: $\theta = [\mu_1, \ldots, \mu_K, \pi_1, \ldots, \pi_K, \Sigma_1, \ldots, \Sigma_K]$

Goal: $\widehat{\theta}_n = \arg\max_{\theta} \log P(D|\theta)$

General EM algorithm

Other Examples: Hidden Markov Models



Observed data: $D = \{x_1, \dots, x_n\}$

Unknown variables: $y = (y_1, \dots, y_n)$

Paramaters: θ $\qquad \theta = [\pi_1, \dots, \pi_K, A, B]$

Initial probabilities: $P(x_1 = i) = \pi_i, \; i = 1, \dots, K$

Transition probabilities: $P(y_{t+1} = j | y_t = i) = A_{ij}$

Emission probabilities: $P(x_{t+1} = l | x_t = i) = B_{il}$

Goal:

$$\widehat{\theta}_n = \arg\max_{\theta} \log P(D|\theta) = \arg\max_{\pi, A, B} \log P(x_1, \dots, x_n|\theta)$$

General EM algorithm

Goal: $\arg\max_\theta \log P(D|\theta)$

$$\log P(D|\theta^t) = \int dy\, q(y) log P(D|\theta^t)$$

$$= \int dy\, q(y) log \left[\frac{P(y,D|\theta^t)}{P(y|D,\theta^t)}\frac{q(y)}{q(y)}\right] \quad \text{since } P(y,D|\theta^t) = P(D|\theta^t)P(y|D,\theta^t)$$

$$= \underbrace{\int dy\, q(y) log P(y,D|\theta^t) \underbrace{- \int dy\, q(y) \log q(y)}_{H(q)}}_{\text{Free energy: } F_{\theta^t}(q(\cdot),D)} + \underbrace{\int dy\, q(y) \log \frac{q(y)}{P(y|D,\theta^t)}}_{KL(q(y)\|P(y|D,\theta^t))}$$

E Step: $Q(\theta^t|\theta^{t-1}) = \mathbb{E}_y[\log P(y,D|\theta^t)|D,\theta^{t-1}]$

$$= \int dy\, P(y|D,\theta^{t-1}) \log P(y,D|\theta^t)$$

M Step: $\theta^t = \arg\max_\theta Q(\theta|\theta^{t-1})$

General EM algorithm

$$\log P(D|\theta^t) = \underbrace{\int dy\, q(y) logP(y, D|\theta^t) \underbrace{- \int dy\, q(y) \log q(y)}_{H(q)}}_{\text{Free energy: } F_{\theta^t}(q(\cdot), D)} + \underbrace{\int dy\, q(y) \log \frac{q(y)}{P(y|D,\theta^t)}}_{KL(q(y)\|P(y|D,\theta^t))}$$

E Step: $Q(\theta^{t+1}|\theta^t) = \int dy\, P(y|D,\theta^t) \log P(y, D|\theta^{t+1})$

Let $q(y) = P(y|D,\theta^t)$

$\Rightarrow KL(q(y)\|P(y|D,\theta^t)) = 0$

$$\Rightarrow \log P(D|\theta^t) = F_{\theta^t}(P(y|D,\theta^t), D)$$

$$= \int dy\, P(y|D,\theta^t) logP(y, D|\theta^t) - \int dy\, P(y|D,\theta^t) \log P(y|D,\theta^t)$$

M Step: $$\leq \int dy\, P(y|D,\theta^t) logP(y, D|\theta^{t+1}) - \int dy\, P(y|D,\theta^t) \log P(y|D,\theta^t)$$

We maximize only here in θ!!!

General EM algorithm

$$\log P(D|\theta^t) = \underbrace{\int dy\, q(y) log P(y, D|\theta^t) \underbrace{- \int dy\, q(y) \log q(y)}_{H(q)}}_{\text{Free energy: } F_{\theta^t}(q(\cdot), D)} + \underbrace{\int dy\, q(y) \log \frac{q(y)}{P(y|D, \theta^t)}}_{KL(q(y) \| P(y|D, \theta^t))}$$

Theorem: During the EM algorithm the marginal likelihood is not decreasing!

$$P(D|\theta^t) \leq P(D|\theta^{t+1})$$

Proof:

$$\begin{aligned}
\log P(D|\theta^t) &= F_{\theta^t}(P(y|D, \theta^t), D) \\
&\leq \int dy\, P(y|D, \theta^t) log P(y, D|\theta^{t+1}) - \int dy\, P(y|D, \theta^t) \log P(y|D, \theta^t) \\
&= F_{\theta^{t+1}}(P(y|D, \theta^t), D) \\
&= \log P(D|\theta^{t+1}) - KL(P(y|D, \theta^t) \| P(y|D, \theta^{t+1})) \\
&\leq \log P(D|\theta^{t+1})
\end{aligned}$$

General EM algorithm

Goal: $\arg\max_{\theta} \log P(D|\theta)$

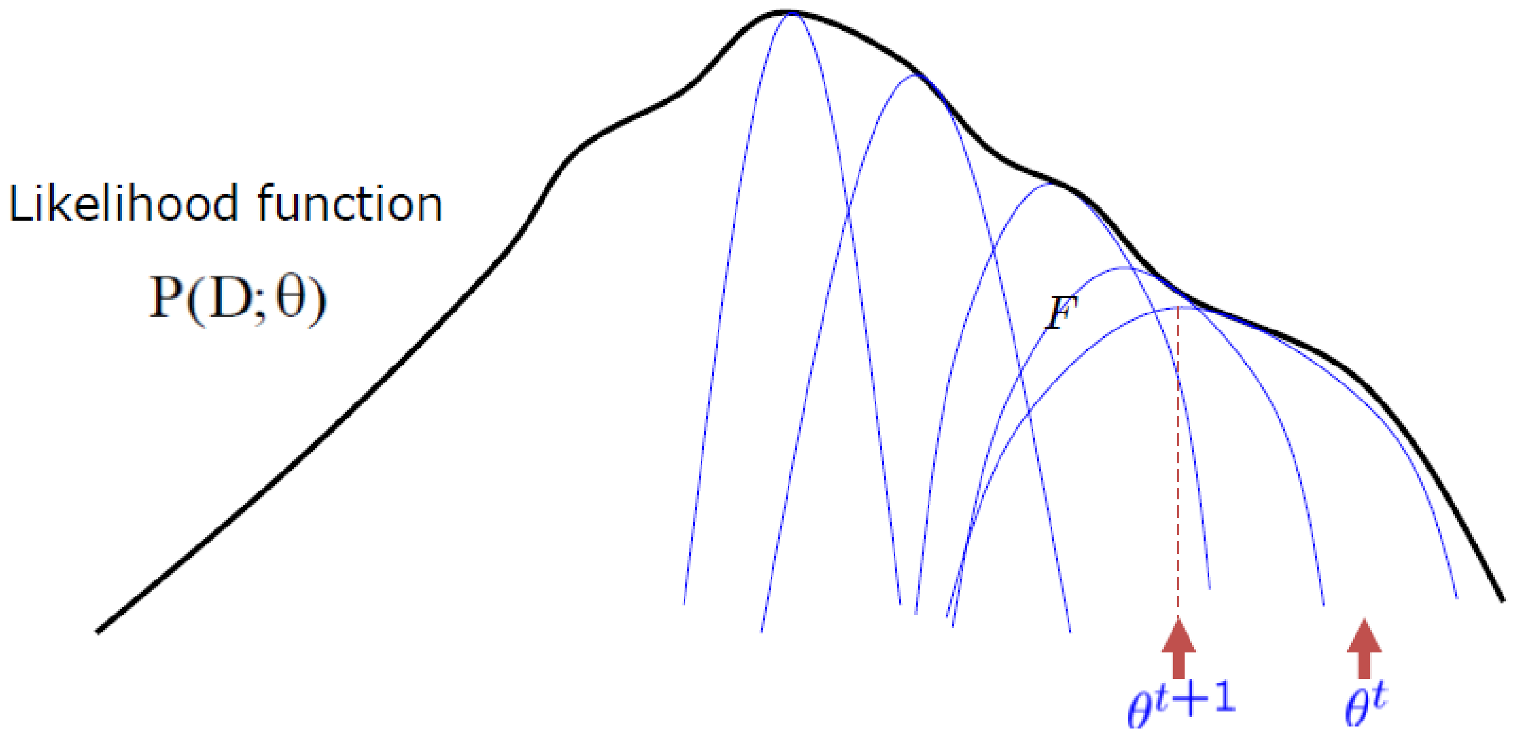
E Step: $$Q(\theta^t|\theta^{t-1}) = \mathbb{E}_y[\log P(y, D|\theta^t)|D, \theta^{t-1}]$$
$$= \int dy\, P(y|D, \theta^{t-1}) \log P(y, D|\theta^t)$$

M Step: $$\theta^t = \arg\max_{\theta} Q(\theta|\theta^{t-1})$$

During the EM algorithm the marginal likelihood is not decreasing!

$$P(D|\theta^t) \leq P(D|\theta^{t+1})$$

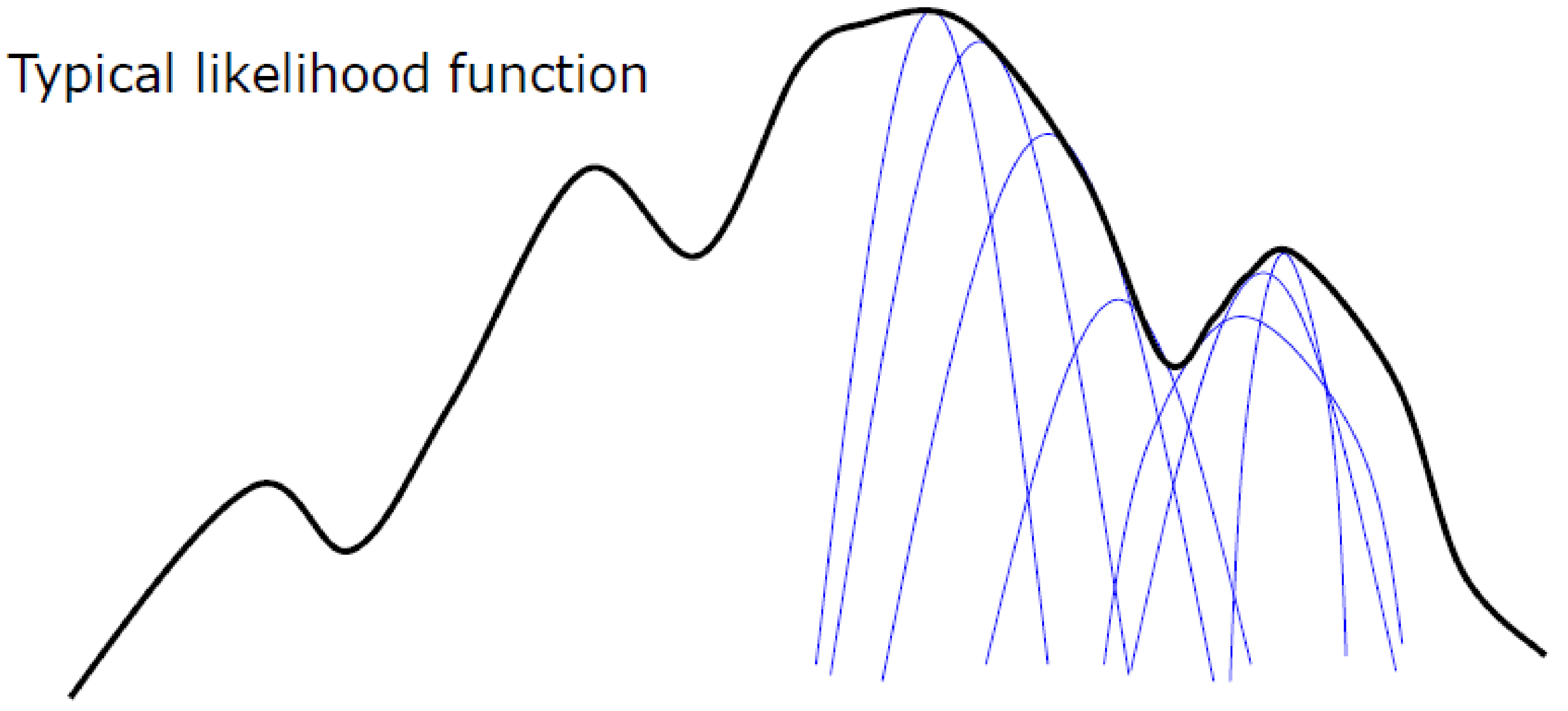
Convergence of EM



Sequence of EM lower bound F-functions

EM monotonically converges to a local maximum of likelihood !

Convergence of EM



Different sequence of EM lower bound F-functions depending on initialization

Use multiple, randomized initializations in practice

Variational Methods

Variational methods

$$\log P(D|\theta^t) = \underbrace{\int dy\, q(y) log P(y, D|\theta^t) - \underbrace{\int dy\, q(y) \log q(y)}_{H(q)}}_{\text{Free energy: } F_{\theta^t}(q(\cdot), D)} + \underbrace{\int dy\, q(y) \log \frac{q(y)}{P(y|D, \theta^t)}}_{KL(q(y) \| P(y|D, \theta^t))}$$

$$\log P(D|\theta^t) \geq F_{\theta^t}(q(\cdot), D)$$

If $P(y|D, \theta^t))$ is complicated, then instead of setting

$$q(y) = P(y|D, \theta^t)),$$

try to find suboptimal maximum points of the free energy.

Variational methods might decrease the marginal likelihood!

Variational methods

$$\log P(D|\theta^t) = \underbrace{\int dy\, q(y) log P(y, D|\theta^t) - \underbrace{\int dy\, q(y) \log q(y)}_{H(q)}}_{\text{Free energy: } F_{\theta^t}(q(\cdot), D)} + \underbrace{\int dy\, q(y) \log \frac{q(y)}{P(y|D, \theta^t)}}_{KL(q(y) \| P(y|D, \theta^t))}$$

$$\log P(D|\theta^t) = F_{\theta^t}(q(\cdot), D) + KL(q(y) \| P(y|D, \theta^t)) \quad \log P(D|\theta^t) \geq F_{\theta^t}(q(\cdot), D)$$

Partial E Step:

θ^t is fixed

$$q^t(\cdot) = \arg\max_{q(\cdot)} F_{\theta^t}(q(\cdot), D) = \arg\min_{q(\cdot)} KL(q(y) \| P(y|D, \theta^t))$$

But **not** necessarily the best max/min which would be $P(y|D, \theta^t))$

Partial M Step:

q^t is fixed

$$\theta^{t+1} = \arg\max_{\theta} F_{\theta}(q^t(\cdot), D)$$

Variational methods might decrease the marginal likelihood!

Summary: EM Algorithm

A way of maximizing likelihood function for hidden variable models.

Finds MLE of parameters when the original (hard) problem can be broken up into two (easy) pieces:

1. Estimate some “missing” or “unobserved” data from observed data and current parameters.

2. Using this “complete” data, find the MLE parameter estimates.

Alternate between filling in the latent variables using the best guess (posterior) and updating the parameters based on this guess:

E Step: $q^t = \arg\max_q F_{\theta^t}(q(\cdot), D)$

M Step: $\theta^{t+1} = \arg\max_\theta F_\theta(q^t(\cdot), D)$

In the M-step we optimize a lower bound F on the likelihood L.

In the E-step we close the gap, making bound F =likelihood L.

EM performs coordinate ascent on F, can get stuck in local optima.