

EM Algorithm

Chapter 9, Statistical learning methods

Yu Hao

March 12, 2019

NaMI, Tongji University

Table of contents ¹

1. K-means
2. Gaussian Mixture Model
3. EM Algorithm

¹Most of the content comes from Aarti Singh,
<https://www.cs.cmu.edu/~epxing/Class/10701-10s/Lecture/lecture10.pdf>

K-means

K-means recap

- Randomly initialize K centers
 - ▷ $\boldsymbol{\mu}^{(0)} = \boldsymbol{\mu}_1^{(0)}, \dots, \boldsymbol{\mu}_K^{(0)}$
- **Classify:** Assign each point $j \in \{1, \dots, N\}$ to nearest center:
 - ▷ $C^{(t)}(j) \leftarrow \arg \min_i \|\boldsymbol{\mu}_i - \mathbf{x}_j\|^2$
- **Recenter:** $\boldsymbol{\mu}_i$ becomes centroid of its points
 - ▷ $\boldsymbol{\mu}_i^{(t+1)} \leftarrow \arg \min_{\boldsymbol{\mu}} \sum_{j: C(j)=i} \|\boldsymbol{\mu} - \mathbf{x}_j\|^2$
 - ▷ Equivalent to $\boldsymbol{\mu}_i \leftarrow$ average of its points!

What is K-means optimizing?

- Potential function $F(\mu, C)$ of centers μ and point allocations C :

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

- Optimal K-means:
 - ▷ $\min_{\mu} \min_C F(\mu, C)$

K-means algorithm

- Optimize potential function:

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

- K-means algorithm:

▷ (1) Fix μ , optimize C

$$\min_C \sum_{j=1}^N \|\mu_{C(j)} - x_j\|^2 = \sum_{j=1}^N \min_{C(j)} \|\mu_{C(j)} - x_j\|^2$$

Exactly first step: assign each point to the **nearest** cluster center

K-means algorithm

- Optimize potential function:

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

- K-means algorithm:

▷ (2) Fix C , optimize μ

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

Exactly second step: average of points in cluster i

K-means algorithm

- Optimize potential function:

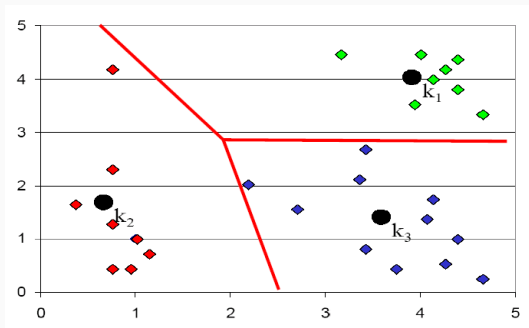
$$\min_{\mu} \min_C F(\mu, C) = \min_{\mu} \min_C \sum_{i=1}^K \sum_{j: C(j)=i} \|\mu_i - x_j\|^2$$

- K-means algorithm:
 - ▷ (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

Iterations of K-means ²

²gif source: wiki https://en.wikipedia.org/wiki/K-means_clustering

K-means decision boundaries



"Linear"
Decision
Boundaries

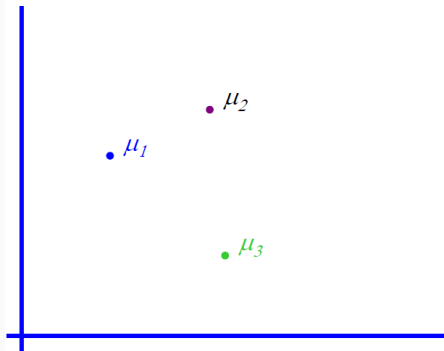
- **Generative Model:**

Assume data comes from a mixture of K Gaussian distributions with same variance.

K-means: Generative model

Mixture of K Gaussian distributions: (Multi-model distribution)

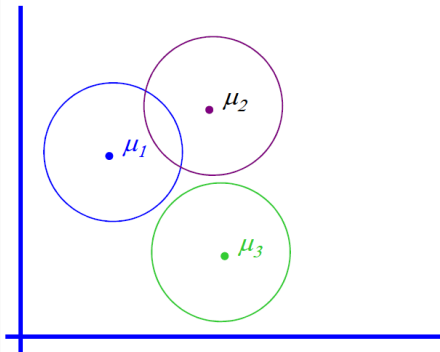
- There are K components
- Component i has an associated mean vector μ_i



K-means: Generative model

Mixture of K Gaussian distributions: (Multi-model distribution)

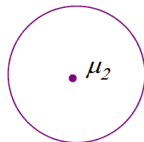
- There are K components
- Component i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 \mathbf{I}$



K-means: Generative model

Mixture of K Gaussian distributions: (Multi-model distribution)

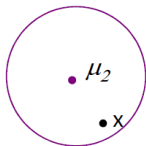
- Each data point is generated according to the following recipe:
- (1) Pick a component at random: choose component i with probability $P(y = i)$



K-means: Generative model

Mixture of K Gaussian distributions: (Multi-model distribution)

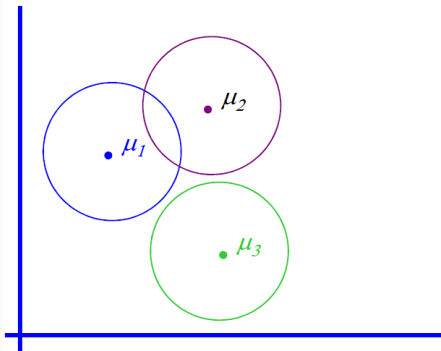
- Each data point is generated according to the following recipe:
- (1) Pick a component at random: choose component i with probability $P(y = i)$
- (2) Data point $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_i, \sigma^2 \mathbf{I})$



K-means: Generative model

Mixture of K Gaussian distributions: (Multi-model distribution)

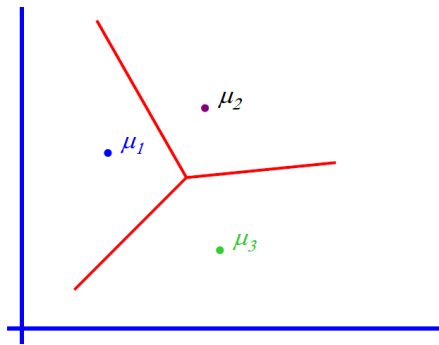
- $p(\mathbf{x}|y = i) \sim \mathcal{N}(\boldsymbol{\mu}_i, \sigma^2 \mathbf{I})$
- $p(\mathbf{x}) = \sum_i p(\mathbf{x}|y = i)P(y = i)$
 - ▷ Mixture component
 - ▷ Mixture proportion



K-means: Generative model

Mixture of K Gaussian distributions: (Multi-model distribution)

- $p(\mathbf{x}|y = i) \sim \mathcal{N}(\boldsymbol{\mu}_i, \sigma^2 \mathbf{I})$
- Gaussian Bayes Classifier:
- "Linear Decision boundary" why?



Gaussian Bayes Classifier Decision Boundary ³

- $p(\mathbf{x}|y = i) \sim \mathcal{N}(\boldsymbol{\mu}_i, \sigma^2 \mathbf{I})$: the covariance is shared between classes,

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

$$\begin{aligned} & \log \pi_i - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T \Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu}_i) \\ = & \log \pi_j - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_j)^T \Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu}_j) \end{aligned}$$

$$\begin{aligned} & C + \mathbf{x}^T \Sigma^{-1} \mathbf{x} - 2\boldsymbol{\mu}_i^T \Sigma^{-1} \mathbf{x} + \boldsymbol{\mu}_i^T \Sigma^{-1} \boldsymbol{\mu}_i \\ = & \mathbf{x}^T \Sigma^{-1} \mathbf{x} - 2\boldsymbol{\mu}_j^T \Sigma^{-1} \mathbf{x} + \boldsymbol{\mu}_j^T \Sigma^{-1} \boldsymbol{\mu}_j \end{aligned}$$

$$\left[2(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^T \Sigma^{-1} \right] \mathbf{x} - (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^T \Sigma^{-1} (\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) = C$$

$$\Rightarrow \mathbf{a}^T \mathbf{x} - b = 0$$

³Mengye Ren,

K-means: MLE

- Maximum Likelihood Estimate (MLE)

$$\arg \max_{\mu, \sigma^2, P(y)} \prod_i P(y_i, \mathbf{x}_i)$$

But we don't know y_i is!

- Maximize marginal likelihood:

$$\begin{aligned} & \arg \max \prod_j P(\mathbf{x}_j) \\ &= \arg \max \prod_j \sum_i^K P(y_j = i, \mathbf{x}_j) \\ &= \arg \max \prod_j \sum_i^K P(y_j = i) p(\mathbf{x}_j | y_j = i) \end{aligned}$$

K-means: MLE

- Maximize marginal likelihood:

$$\begin{aligned} & \arg \max \prod_j P(\mathbf{x}_j) \\ &= \arg \max \prod_j \sum_i^K P(y_j = i, \mathbf{x}_j) \\ &= \arg \max \prod_j \sum_i^K P(y_j = i) p(\mathbf{x}_j | y_j = i) \end{aligned}$$

- Substitute with Gaussian distribution probability:

$$P(y_j = i, \mathbf{x}_j) \propto P(y_j = i) \exp \left[-\frac{1}{2\sigma^2} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2 \right]$$

K-means: MLE

- If each \mathbf{x}_j belongs to one class $C(j)$ (hard assignment), marginal likelihood:

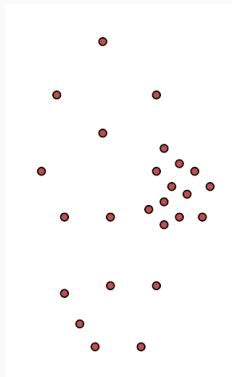
$$P(y_j = i) = \begin{cases} 1 & C(j) = i \\ 0 & \text{else} \end{cases}$$

- Then, the log-likelihood function is

$$\begin{aligned} \ln \prod_{j=1}^N \sum_{i=1}^K P(y_j = i, \mathbf{x}_j) &\propto \ln \prod_{j=1}^N \exp \left[-\frac{1}{2\sigma^2} \|\mathbf{x}_j - \boldsymbol{\mu}_{C(j)}\|^2 \right] \\ &= \sum_{j=1}^N -\frac{1}{2\sigma^2} \|\mathbf{x}_j - \boldsymbol{\mu}_{C(j)}\|^2 \end{aligned}$$

Same as K-means!

One bad case for K-means



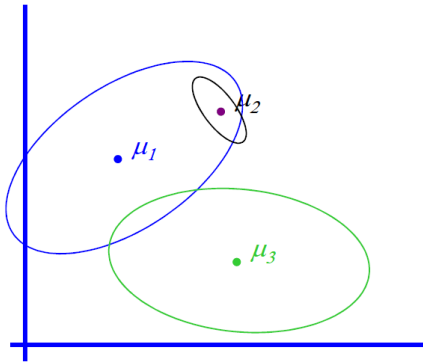
- Clusters may not be linear separable
- Clusters may overlap
- Some clusters may be "wider" than others

Gaussian Mixture Model

General GMM

GMM-Gaussian Mixture Model (Multi-model distribution)

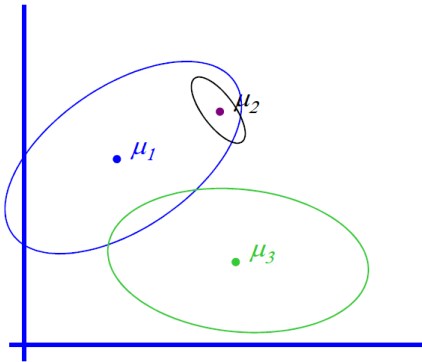
- There are K components
- Component i has an associated mean vector μ_i
- Each component generates data from Gaussian with mean μ_i and covariance matrix Σ_i



General GMM

GMM-Gaussian Mixture Model (Multi-model distribution)

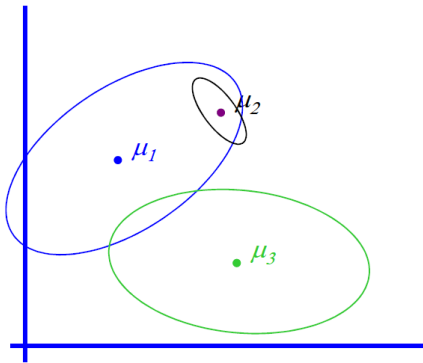
- Each data is generated according to the following recipe:
- (1) Pick a component at random:
Choose component i with probability $P(y = i)$
- (2) Data point $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$



General GMM

GMM-Gaussian Mixture Model (Multi-model distribution)

- $p(\mathbf{x}|y = i) \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$
- $p(\mathbf{x}) = \sum_i p(\mathbf{x}|y = i)P(y = i)$
 - ▷ Mixture component
 - ▷ Mixture proportion



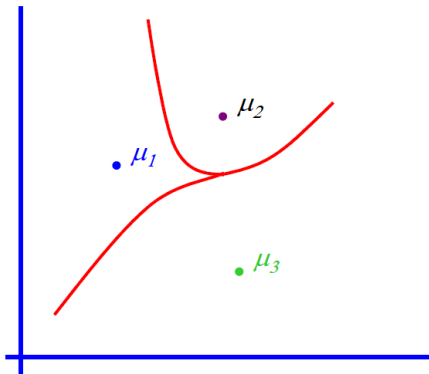
General GMM

GMM-Gaussian Mixture Model (Multi-model distribution)

- $p(\mathbf{x}|y = i) \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$
- Gaussian Bayes Classifier:

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

$$\Rightarrow \mathbf{x}^T Q \mathbf{x} - 2\mathbf{b}^T \mathbf{x} + b = 0$$



"Quadratic Decision boundary"
second-order terms don't cancel

GMM: marginal likelihood

- Maximize marginal likelihood:

$$\begin{aligned} & \arg \max \prod_j P(\mathbf{x}_j) \\ &= \arg \max \prod_j \sum_i^K P(y_j = i, \mathbf{x}_j) \\ &= \arg \max \prod_j \sum_i^K P(y_j = i) p(\mathbf{x}_j | y_j = i) \end{aligned}$$

GMM: marginal likelihood

- Uncertain about class of each \mathbf{x}_j (soft assignment),

$$P(y_j = i) = P(y = i)$$

$$\prod_{j=1}^N \sum_{i=1}^K P(y_j = i, \mathbf{x}_j) \propto$$

$$\prod_{j=1}^N \sum_{i=1}^K P(y = i) \frac{1}{\sqrt{\det(\boldsymbol{\Sigma}_i)}} \exp \left[-\frac{1}{2} (\mathbf{x}_j - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i) \right]$$

- How do we find the $\boldsymbol{\mu}_i$'s which give max marginal likelihood?

- ▷ Set $\frac{\partial F}{\partial \boldsymbol{\mu}_i} = 0$ and solve for $\boldsymbol{\mu}_i$'s. Non-linear non-analytically solvable.
- ▷ Use gradient decent: Often slow but doable.

EM Algorithm

Expectation-Maximization (EM)

- EM is an optimization strategy for objective functions that can be interpreted as likelihoods in the presence of missing data.
- It is much simpler than gradient methods.
- EM is an iterative algorithm with two linked steps:
 - ▷ E-step: fill in hidden values using inference
 - ▷ M-step: apply standard MLE methods
- This procedure monotonically improves the likelihood. Thus it always converges to a local optimum of the likelihood.

EM: A simple case

- We have unlabeled data $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$
- We know there are K classes
- We know $P(y = 1), P(y = 2), \dots, P(y = K)$
- We **don't** know $\mu_1, \mu_2, \dots, \mu_K$
- We know common variance σ^2

EM: A simple case

- Problem formulation:

$$\begin{aligned} & P(\text{data} | \mu_1 \dots \mu_K) \\ &= P(\mathbf{x}_1 \dots \mathbf{x}_N | \mu_1 \dots \mu_K) \\ &= \prod_{j=1}^N p(\mathbf{x}_j | \mu_1 \dots \mu_K) \quad \text{Independent data} \\ &= \prod_{j=1}^N \sum_{i=1}^K p(\mathbf{x}_j | \mu_i) P(y = i) \quad \text{Marginalize over class} \\ &\propto \prod_{j=1}^N \sum_{i=1}^K \exp\left(-\frac{1}{2\sigma^2} \|\mathbf{x}_j - \mu_i\|^2\right) P(y = i) \end{aligned}$$

Expectation (E) step

- If we know μ_1, \dots, μ_K , then easily compute probability about point \mathbf{x}_j belongs to class $y = i$

$$P(y = i | \mathbf{x}_j, \mu_1, \dots, \mu_K) \propto \exp \left(-\frac{1}{2\sigma^2} \|\mathbf{x}_j - \mu_i\|^2 \right) P(y = i)$$

Maximization (M) step

- If we know probability about point \mathbf{x}_j belongs to class $y = i$, then MLE for μ_i is weighted average.
- Imagine multiple copies of each \mathbf{x}_j , each with weight $P(y = i|\mathbf{x}_j)$:

$$\mu_i = \frac{\sum_{j=1}^N P(y = i|\mathbf{x}_j) \mathbf{x}_j}{\sum_{j=1}^N P(y = i|\mathbf{x}_j)}$$

EM for spherical, same variance GMMs

- E-step

$$P(y = i | \mathbf{x}_j, \boldsymbol{\mu}_1 \dots \boldsymbol{\mu}_K) \propto \exp \left(-\frac{1}{2\sigma^2} \|\mathbf{x}_j - \boldsymbol{\mu}_i\|^2 \right) P(y = i)$$

- ▷ Compute "expected" classes of all data points for each class
- ▷ In K-means, we do hard assignment; EM dose soft assignment

- M-step

$$\boldsymbol{\mu}_i = \frac{\sum_{j=1}^N P(y = i | \mathbf{x}_j) \mathbf{x}_j}{\sum_{j=1}^N P(y = i | \mathbf{x}_j)}$$

- ▷ Compute Max. like $\boldsymbol{\mu}$ given our data's class membership distributions.

EM for general GMMs

- Iterate. On iteration t let our estimates be

$$\lambda_t = \boldsymbol{\mu}_1^{(t)}, \boldsymbol{\mu}_2^{(t)}, \dots, \boldsymbol{\mu}_K^{(t)}, \boldsymbol{\Sigma}_1^{(t)}, \boldsymbol{\Sigma}_2^{(t)}, \dots, \boldsymbol{\Sigma}_K^{(t)}, p_1^{(t)}, p_2^{(t)}, \dots, p_K^{(t)}$$

▷ $p_i^{(t)}$ is shorthand for estimate of $P(y = i)$

- E-step:

$$P(y = i | \mathbf{x}_j, \lambda_t) \propto p_i^{(t)} p(\mathbf{x}_j | \boldsymbol{\mu}_i^{(t)}, \boldsymbol{\Sigma}_i^{(t)})$$

EM for general GMMs

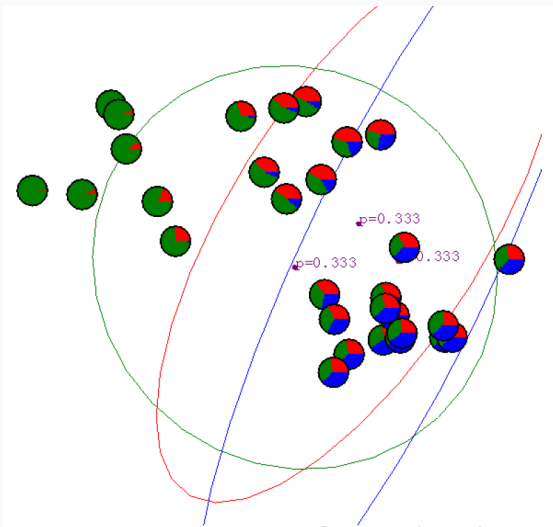
- M-step

$$\mu_i^{(t+1)} = \frac{\sum_{j=1}^N P(y = i | \mathbf{x}_j, \lambda_t) \mathbf{x}_j}{\sum_{j=1}^N P(y = i | \mathbf{x}_j, \lambda_t)}$$

$$\Sigma_i^{(t+1)} = \frac{\sum_j P(y = i | \mathbf{x}_j, \lambda_t) (\mathbf{x}_j - \mu_i^{(t+1)}) (\mathbf{x}_j - \mu_i^{(t+1)})^T}{\sum_j P(y = i | \mathbf{x}_j, \lambda_t)}$$

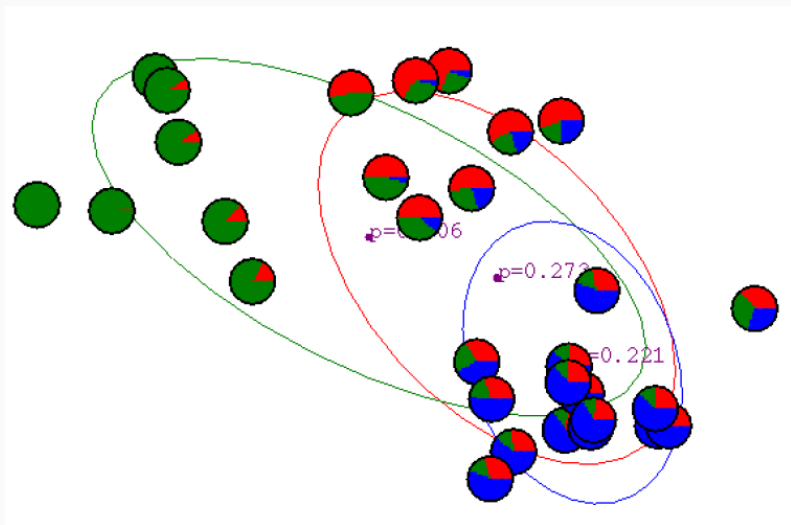
$$p_i^{(t+1)} = \frac{\sum_j P(y = i | \mathbf{x}_j, \lambda_t)}{N}$$

EM for general GMMS: Example



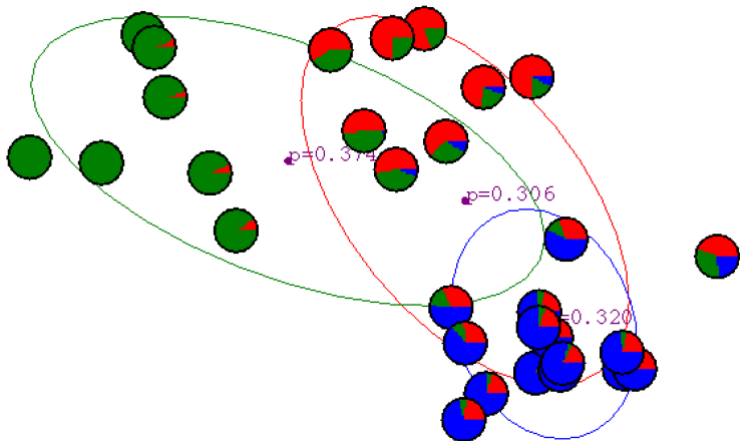
initialization

EM for general GMMS: Example



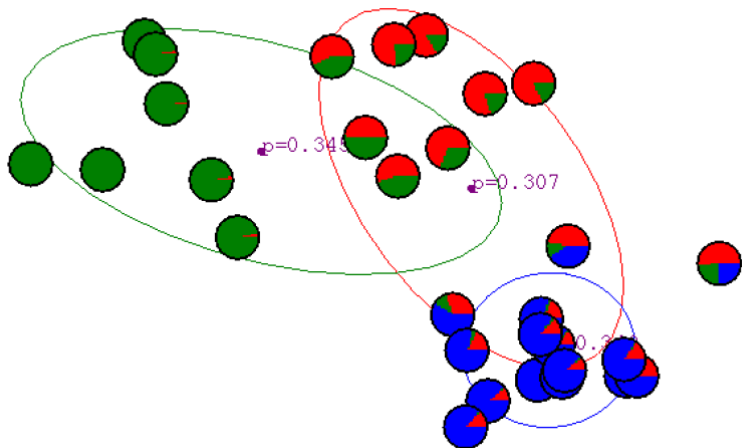
After 1st iteration

EM for general GMMS: Example



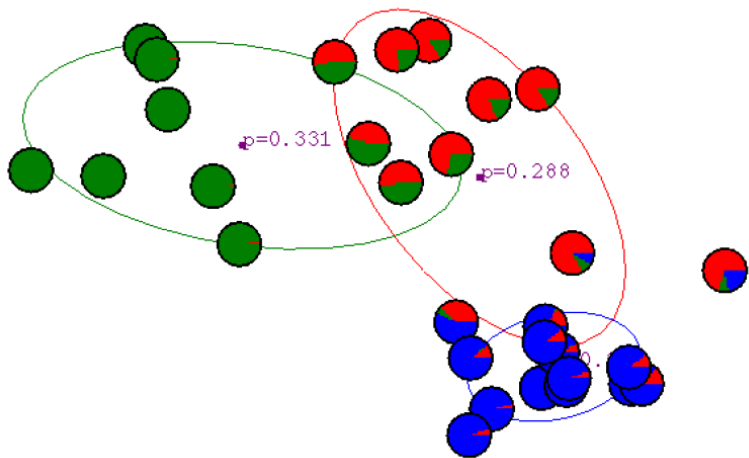
After 2nd iteration

EM for general GMMS: Example



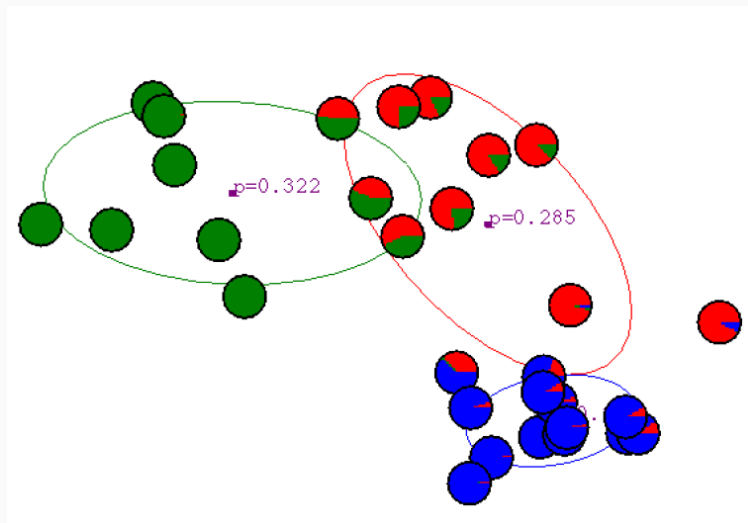
After 3rd iteration

EM for general GMMS: Example



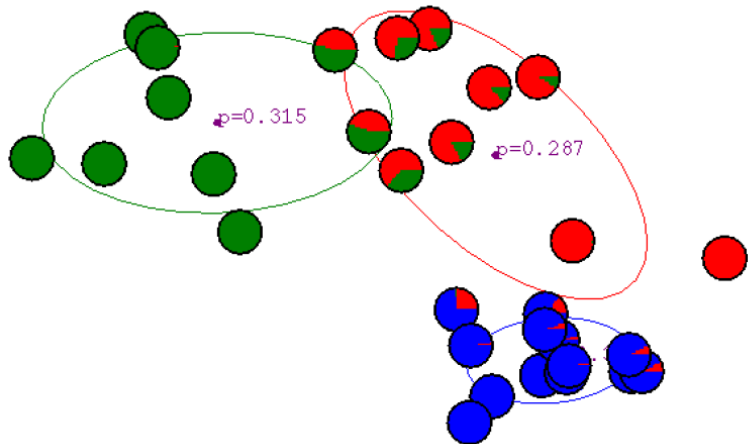
After 4th iteration

EM for general GMMS: Example



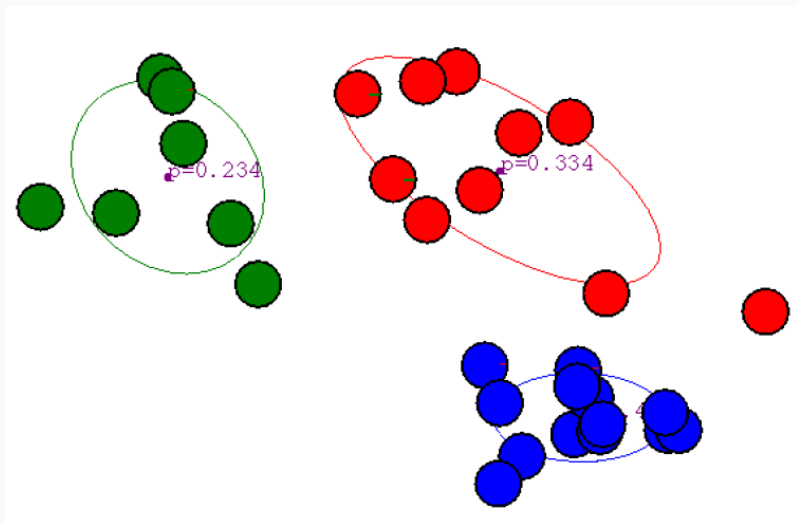
After 5th iteration

EM for general GMMS: Example



After 6th iteration

EM for general GMMS: Example



After 20th iteration

General EM algorithm

- Marginal likelihood: \mathbf{x} is observed, \mathbf{z} is missing:

$$\begin{aligned} P(\mathbf{D}; \theta) &= \log \prod_{j=1}^N P(\mathbf{x}_j | \theta) \\ &= \sum_{j=1}^N \log P(\mathbf{x}_j | \theta) \\ &= \sum_{j=1}^N \log \sum_{\mathbf{z}} P(\mathbf{x}_j, \mathbf{z} | \theta) \end{aligned}$$

E-step

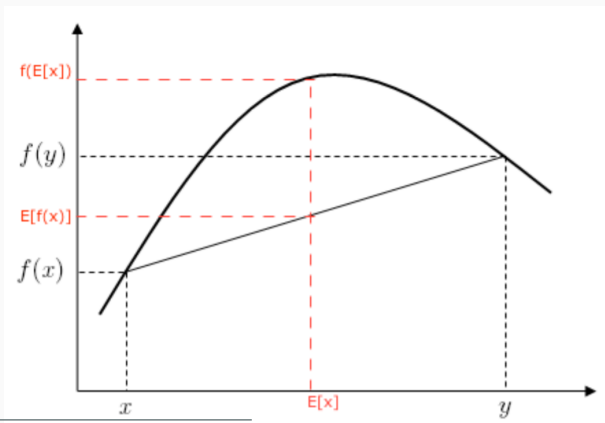
- \mathbf{x} is observed, \mathbf{z} is missing
- Compute probability of missing data given current choice of θ

$$Q^{(t+1)}(\mathbf{z}|\mathbf{x}_j) = P(\mathbf{z}|\mathbf{x}_j, \theta^{(t)})$$

Jensen's inequality⁴

- For a random variable x , if $f(x)$ is concave, then

$$f(E[x]) \geq E[f(x)]$$



⁴wiki, https://en.wikipedia.org/wiki/Jensen%27s_inequality

Lower-bound on marginal likelihood

$$\begin{aligned}P(\mathbf{D}; \theta) &= \sum_{j=1}^N \log \sum_z P(\mathbf{x}_j, z | \theta) \\&= \sum_{j=1}^N \log \sum_z Q(z | \mathbf{x}_j) \frac{P(z, \mathbf{x}_j | \theta)}{Q(z | \mathbf{x}_j)} \\&\geq \sum_{j=1}^N \sum_z Q(z | \mathbf{x}_j) \log \frac{P(z, \mathbf{x}_j | \theta)}{Q(z | \mathbf{x}_j)} \quad \text{Jensen's inequality} \\&= \sum_{j=1}^N \sum_z Q(z | \mathbf{x}_j) \log P(z, \mathbf{x}_j | \theta) + N.H(Q) \quad \text{entropy of } Q\end{aligned}$$

$$P(\mathbf{D}; \theta) \geq \sum_{j=1}^N \sum_z Q(z|x_j) \log P(z, x_j|\theta) + N.H(Q)$$

- Maximize lower bound on marginal likelihood

$$\theta^{(t+1)} \leftarrow \arg \max_{\theta} \sum_{j=1}^N \sum_z Q^{(t+1)}(z|x_j) \log P(z, x_j|\theta)$$

Convergence of EM

$$P(\mathbf{D}; \theta) \geq F(\theta, Q)$$

- **M-step:** Fix Q , maximize F over θ

$$P(\mathbf{D}; \theta) \geq F(\theta, Q^{(T)})$$

$$= \sum_{j=1}^N \sum_z Q^{(t)}(z|x_j) \log P(z, x_j|\theta) + N.H(Q^{(t)})$$

Maximizes lower bound F on marginal likelihood

Convergence of EM

- **E-step:** Fix θ , maximize F over Q

$$P(\mathbf{D}; \theta^{(t)}) \geq F(\theta^{(t)}, Q)$$

$$\begin{aligned} &= \sum_{j=1}^N \sum_z Q(z|x_j) \log \frac{P(z, x_j | \theta^{(t)})}{Q(z|x_j)} \\ &= \sum_{j=1}^N \sum_z Q(z|x_j) \log \frac{P(z|x_j, \theta^{(t)}) P(x_j | \theta^{(t)})}{Q(z|x_j)} \\ &= \sum_{j=1}^N \sum_z Q(z|x_j) \log \frac{P(z|x_j, \theta^{(t)})}{Q(z|x_j)} \quad \leftarrow \text{KL divergence} \\ &+ \sum_{j=1}^N \sum_z Q(z|x_j) \log P(x_j | \theta^{(t)}) \quad \leftarrow P(\mathbf{D}; \theta^{(t)}) \end{aligned}$$

Convergence of EM

- **E-step:** Fix θ , maximize F over Q

$$\begin{aligned} P(\mathbf{D}; \theta^{(t)}) &\geq F(\theta^{(t)}, Q) \\ &= \sum_{j=1}^N -KL(Q(z|\mathbf{x}_j), P(z|\mathbf{x}_j, \theta^{(t)})) + P(\mathbf{D}; \theta^{(t)}) \end{aligned}$$

- $KL \geq 0$ (why?), F is maximized if KL divergence = 0

$$KL(Q, P) = 0 \text{ if } Q = P$$

- Recall E-step:

$$Q^{(t+1)}(z|\mathbf{x}_j) = P(z|\mathbf{x}_j, \theta^{(t)})$$

Convergence of EM

- **M-step:** Fix Q , maximize F over θ

$$P(\mathbf{D}; \theta) \geq F(\theta, Q^{(t)}) = \sum_{j=1}^N \sum_z Q^{(t)}(z|x_j) \log P(z, x_j|\theta) + N.H(Q^{(t)})$$

Maximize lower bound F on marginal likelihood

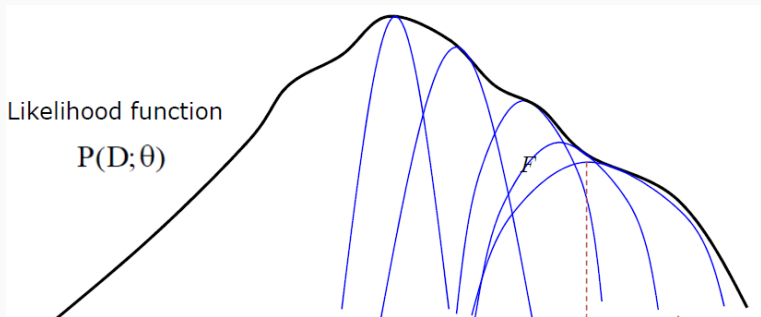
- **E-step:** Fix θ , maximize F over Q

$$P(\mathbf{D}; \theta^{(t)}) \geq F(\theta^{(t)}, Q) = P(\mathbf{D}; \theta^{(t)}) - \sum_{j=1}^N KL \left(Q(z|x_j) || P(z|x_j, \theta^{(t)}) \right)$$

Re-aligns F with marginal likelihood

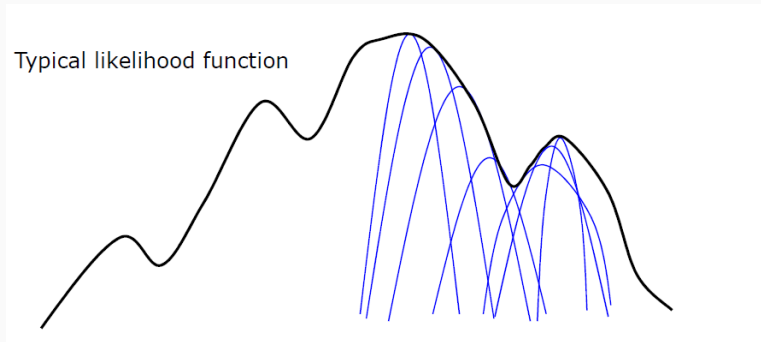
$$F(\theta^{(t)}, Q^{(t+1)}) = P(\mathbf{D}; \theta^{(t)})$$

Monotonic convergence of EM






- EM monotonically converges to a local maximum of likelihood

Monotonic convergence of EM



- Different sequence of EM surrogate F-functions depending on initialization.
- Use multiple, randomized initializations in practice

Recent research papers on EM algorithm

-  Balakrishnan, Sivaraman, Martin J. Wainwright, and Bin Yu. "**Statistical guarantees for the EM algorithm: From population to sample-based analysis.**" The Annals of Statistics 45.1 (2017): 77-120.
-  Schwartz, Boaz, Sharon Gannot, and Emanuël AP Habets. "**Online speech dereverberation using Kalman filter and EM algorithm.**" IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) 23.2 (2015): 394-406.
-  Zhang, Wen, Ye Yang, and Qing Wang. "**Using Bayesian regression and EM algorithm with missing handling for software effort prediction.**" Information and software technology 58 (2015): 58-70.

Thanks