EM Algorithm

Aarti Singh

Slides courtesy: Eric Xing, Carlos Guetrin

Machine Learning 10-701/15-781
Oct 27, 2010

K-means Recap …

- Randomly initialize *k* centers
    - $\mu^{(0)} = \mu_1^{(0)}, \ldots, \mu_k^{(0)}$

- **Classify**: Assign each point j∈{1,…m} to nearest center:
    - $C^{(t)}(j) \leftarrow \arg\min_{i=1,\ldots,k} \|\mu_i^{(t)} - x_j\|^2$

- **Recenter**: μ_i becomes centroid of its points:
    - $\mu_i^{(t+1)} \leftarrow \arg\min_{\mu} \sum_{j:C^{(t)}(j)=i} \|\mu - x_j\|^2 \qquad i \in \{1, \ldots, k\}$
    - Equivalent to $\mu_i \leftarrow$ average of its points!

What is K-means optimizing?

- Potential function F(**μ**,C) of centers **μ** and point allocations C:

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

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

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

K-means algorithm

- Optimize potential function:

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

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

- **K-means algorithm:**

(1) Fix μ, optimize C

$$\min_{C(1),C(2),\ldots,C(m)} \sum_{j=1}^{m} \|\mu_{C(j)} - x_j\|^2 = \sum_{j=1}^{m} \underbrace{\min_{C(j)} \|\mu_{C(j)} - x_j\|^2}_{\text{Exactly first step – assign each point to the nearest cluster center}}$$

K-means algorithm

- Optimize potential function:

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

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

- **K-means algorithm:**

(2) Fix C, optimize **μ**

$$\min_{\mu_1,\mu_2,\ldots\mu_K} \sum_{i=1}^{K} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2 = \sum_{i=1}^{K} \underbrace{\min_{\mu_i} \sum_{j:C(j)=i} \|\mu_i - x_j\|^2}$$

Solution: average of points in cluster i
Exactly second step (re-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:** (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

Partitioning Algorithms

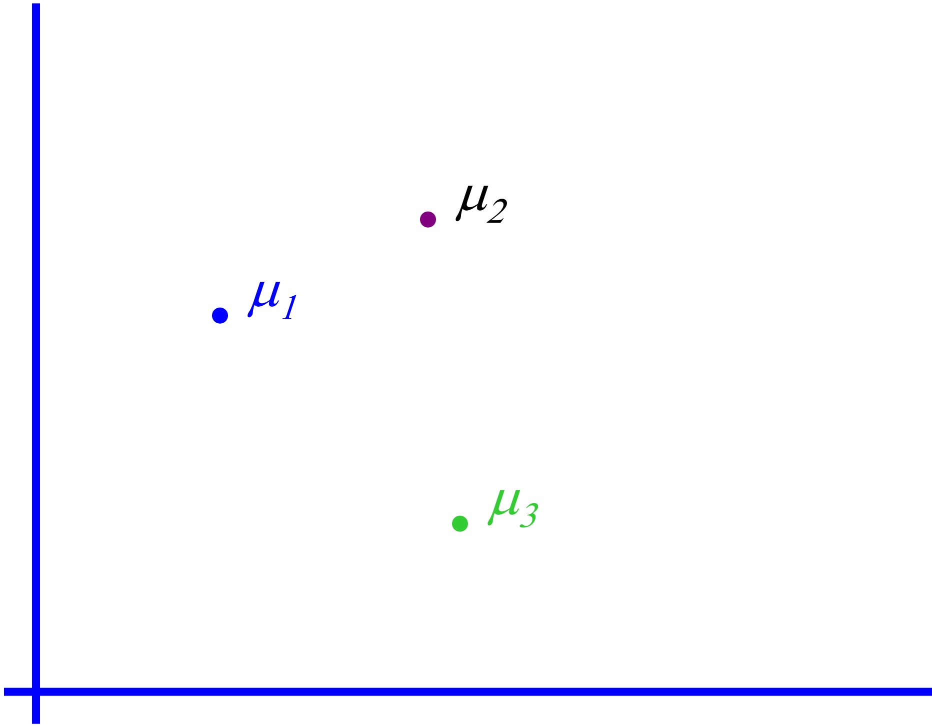
- K-means
  - **hard assignment**: each object belongs to only one cluster
- Mixture modeling
  - **soft assignment**: probability that an object belongs to a cluster

Generative approach

Gaussian Mixture Model

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

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

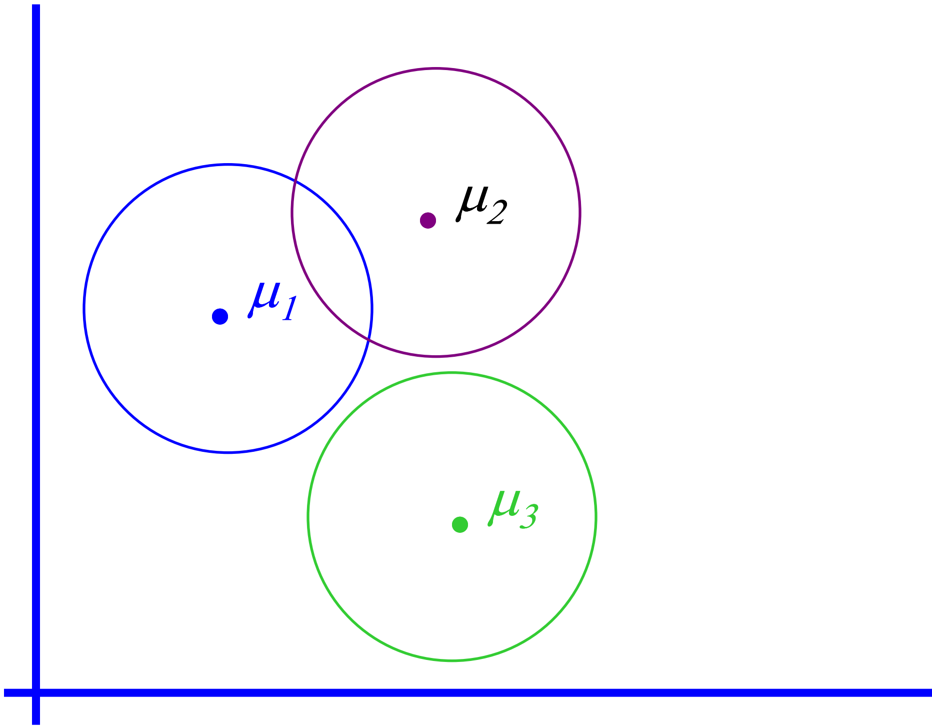


Gaussian Mixture Model

Mixture of K Gaussians distributions: (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 $\sigma^2 I$

Each data point is generated according to the following recipe:



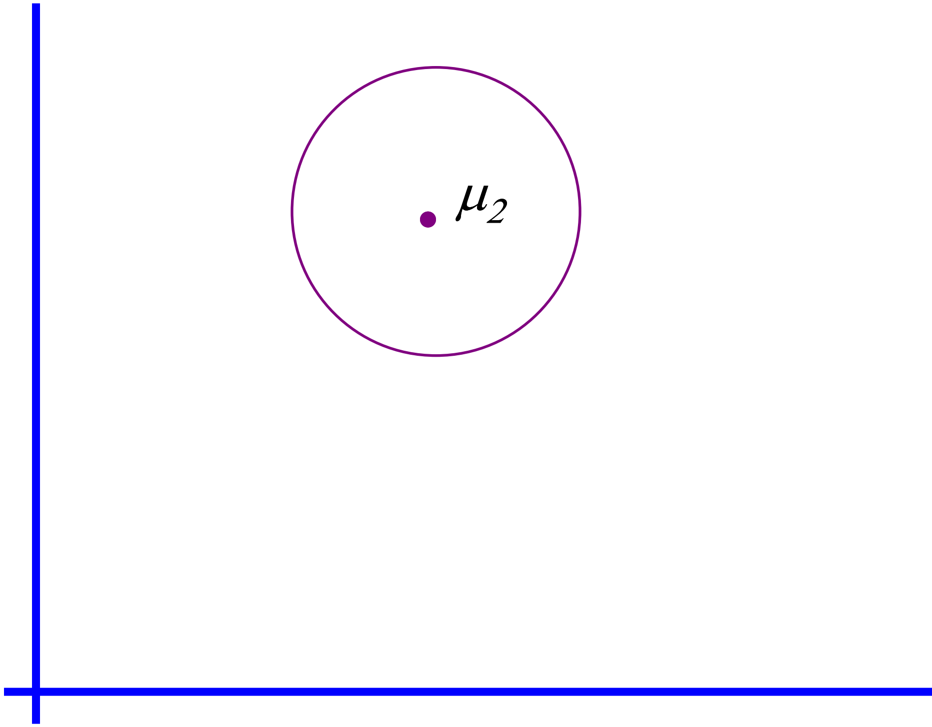
Gaussian Mixture Model

Mixture of K Gaussians distributions: (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 $\sigma^2 I$

Each data point is generated according to the following recipe:

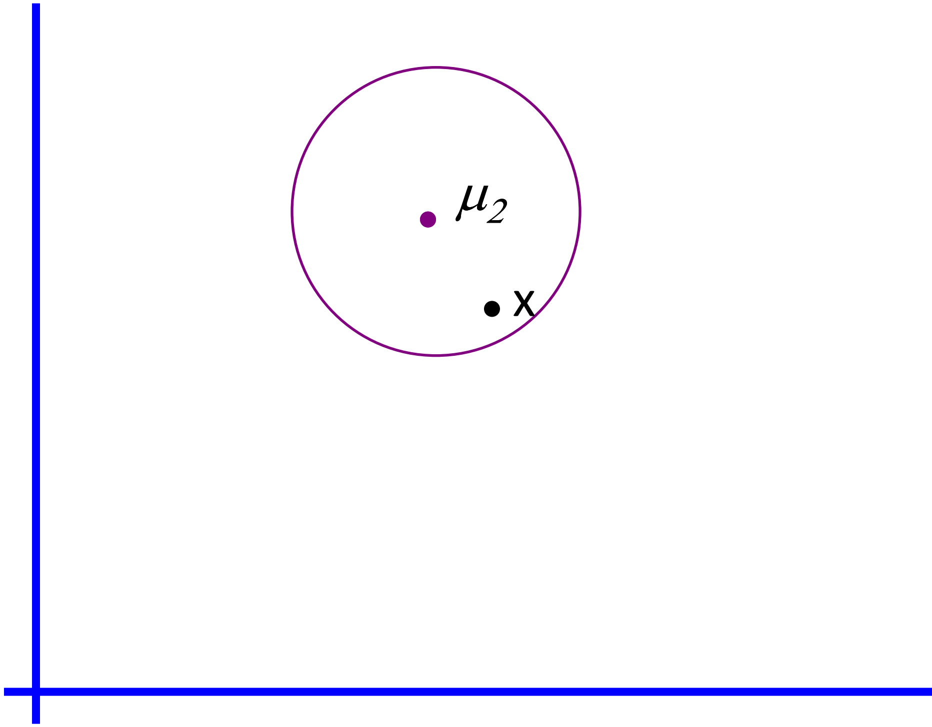
1) Pick a component at random: Choose component i with probability *P(y=i)*



Gaussian Mixture Model

Mixture of K Gaussians distributions: (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 $\sigma^2 I$



Each data point is generated according to the following recipe:

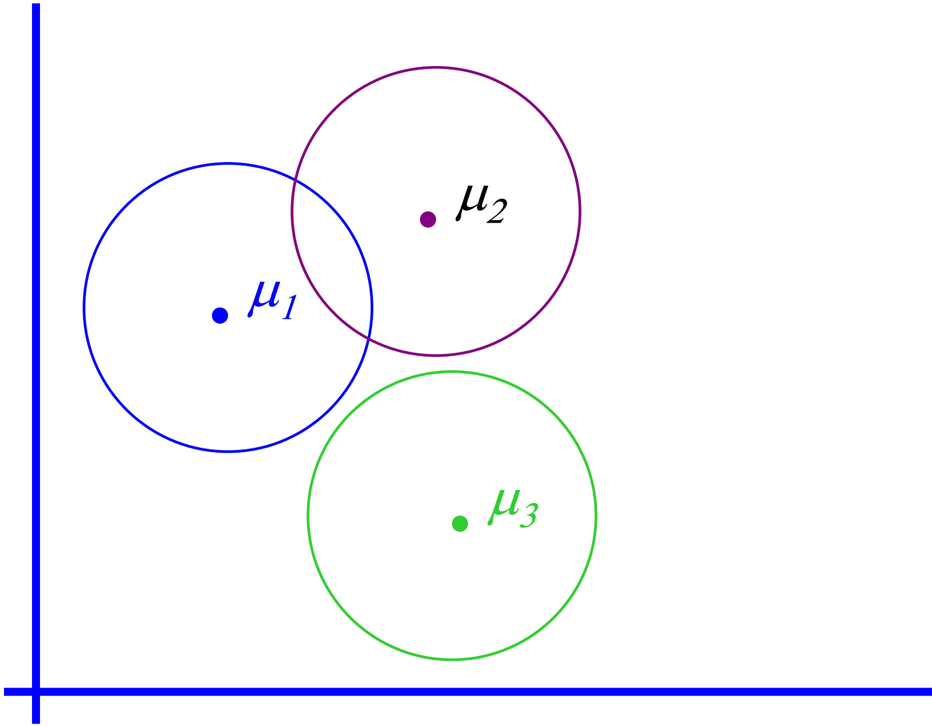
1) Pick a component at random: Choose component i with probability *P(y=i)*
2) Datapoint x ~ $\mathrm{N}(\mu_i, \sigma^2 I)$

Gaussian Mixture Model

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

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

$$p(x) = \sum_i \underbrace{p(x|y=i)}_{\text{Mixture component}} \; \underbrace{P(y=i)}_{\text{Mixture proportion}}$$



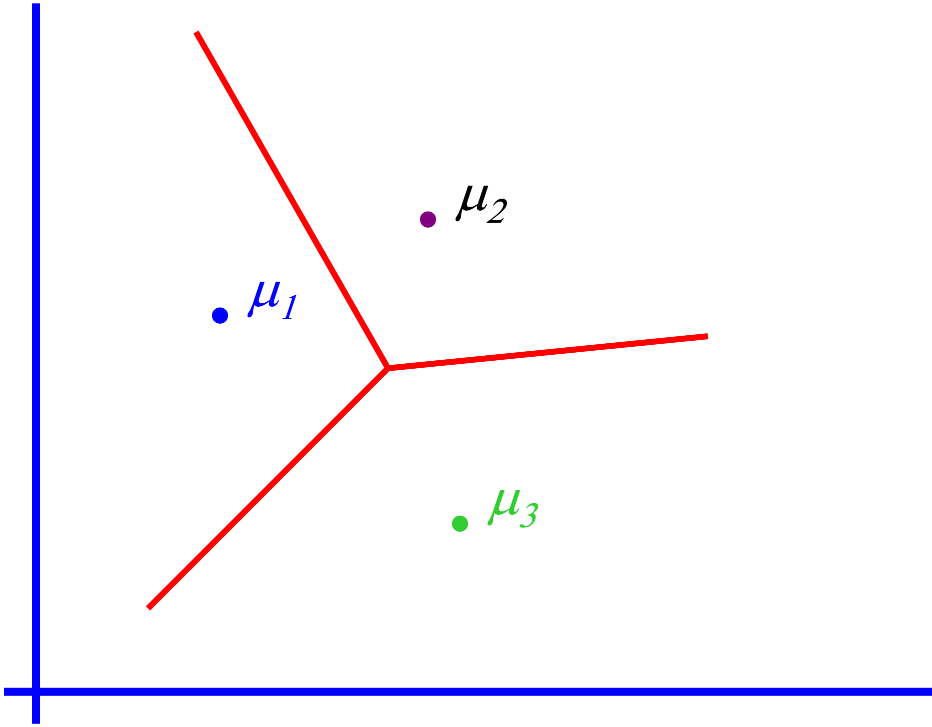
Recall: Gaussian Bayes Classifier

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

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

Gaussian Bayes Classifier:

$$\log \frac{\mathrm{P}(y=i\,|\,x)}{\mathrm{P}(y=j\,|\,x)} = \log \frac{p(x\,|\,y=i)\mathrm{P}(y=i)}{p(x\,|\,y=j)\mathrm{P}(y=j)} = \boxed{w}^{\mathrm{T}} x$$



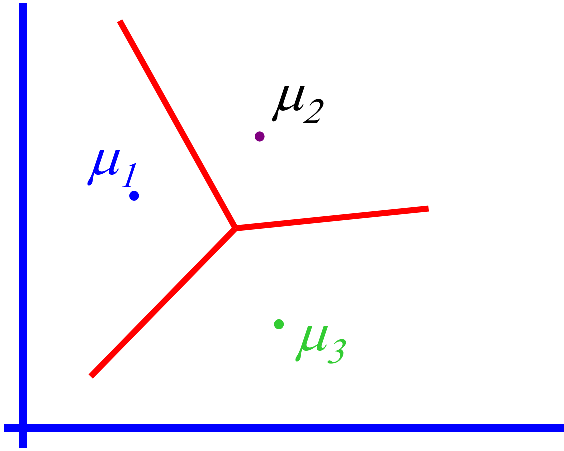
→ Depends on $\mu_1, \mu_2, .. , \mu_K, \sigma^2$, P(y=1),..., P(Y=k)

"Linear Decision boundary" – Recall that second-order terms cancel out

MLE for GMM

Maximum Likelihood Estimate (MLE)

$$\underset{\mu_1,\, \mu_2,\, ..\, ,\, \mu_K, \sigma^2,\; P(y=1),\ldots,\, P(Y=k)}{\text{argmax}} \prod_{j=1}^{m} P(y_j, x_j)$$



But we don't know y_j's!!!

Maximize marginal likelihood:

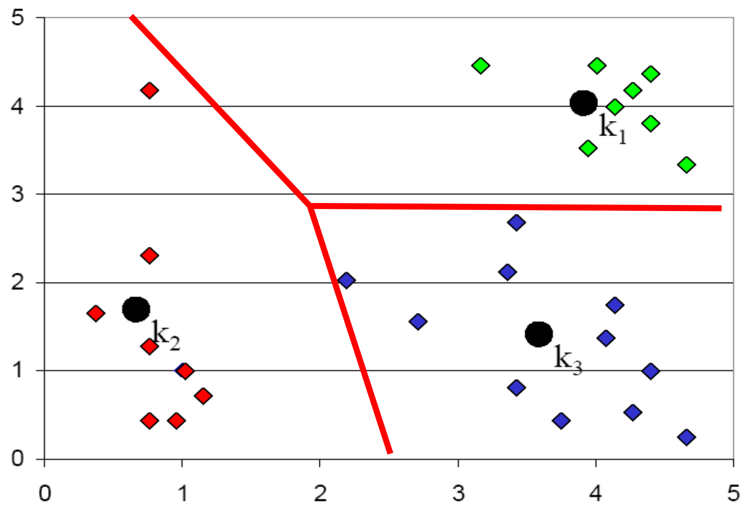
$$\text{argmax} \prod_j P(x_j) = \text{argmax} \prod_j \sum_{i=1}^{K} P(y_j=i, x_j)$$

$$= \text{argmax} \prod_j \sum_i^{K} P(y_j=i) p(x_j | y_j=i)$$

$$= \text{argmax} \prod_j \sum_{i=1}^{K} P(y_j=i) \exp\left[-\frac{1}{2\sigma^2}\left\|x_j - \mu_i\right\|^2\right]$$

K-means and GMM

"Linear" Decision Boundaries



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

Hard assignment:

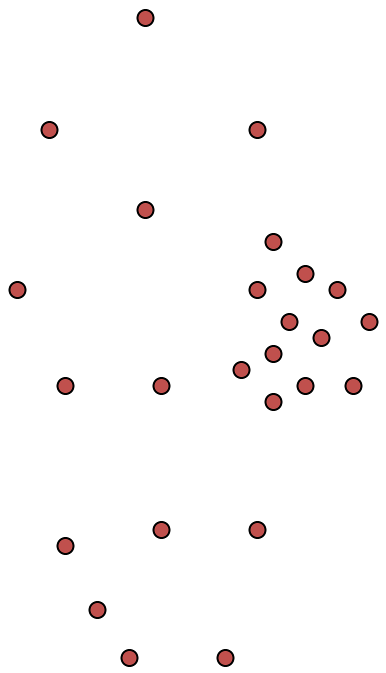
$$P(y_j = i) = 1 \qquad \text{if} \quad i = C(n)$$
$$= 0 \qquad \text{otherwise}$$

Maximize marginal likelihood:

$$\underset{\mu_1, \mu_2, .., \mu_K, \sigma^2, P(y=1), ..., P(y=k)}{\text{argmax}} \prod_j P(x_j) \quad \equiv \underset{\mu_1, .., \mu_K, C(1), ..., C(m)}{\arg\min} \sum_{j=1}^{m} \left\| x_j - \mu_{C(j)} \right\|^2$$

Same as K-means!!!

(One) bad case for K-means

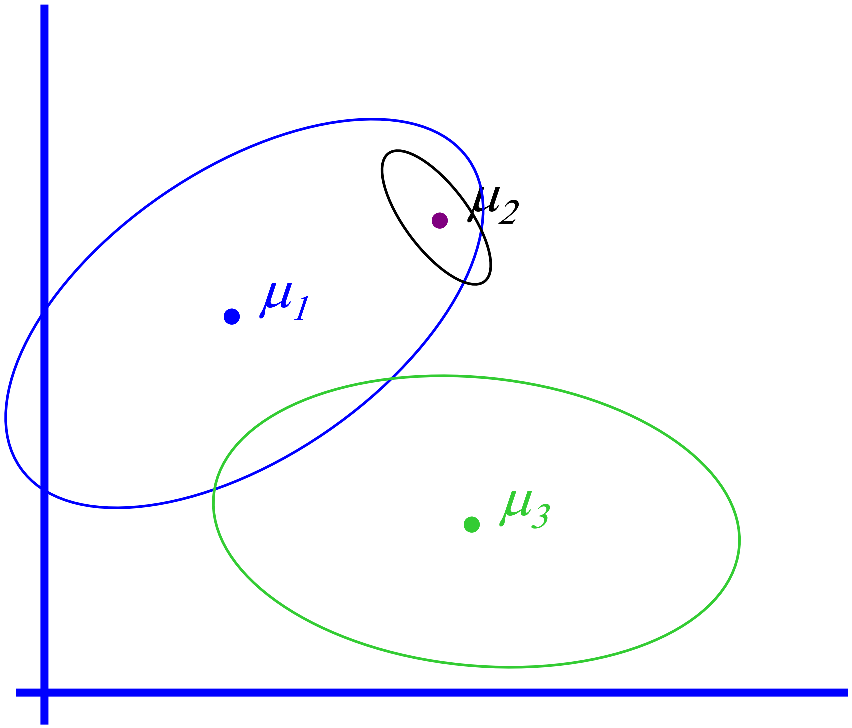


- Clusters may not be linearly separable
- Clusters may overlap
- Some clusters may be “wider” than others

General GMM

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:

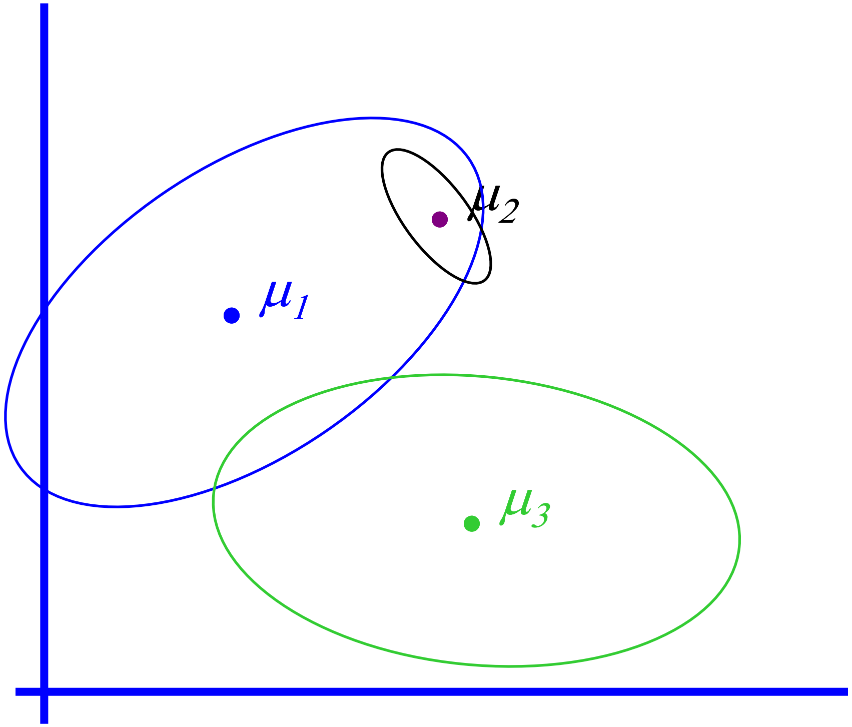
1) Pick a component at random: Choose component i with probability *P(y=i)*
2) Datapoint x ~ $\mathrm{N}(\mu_i, \Sigma_i)$

General GMM

GMM – Gaussian Mixture Model (Multi-modal distribution)

$$p(x|y=i) \sim \mathrm{N}(\mu_i, \Sigma_i)$$

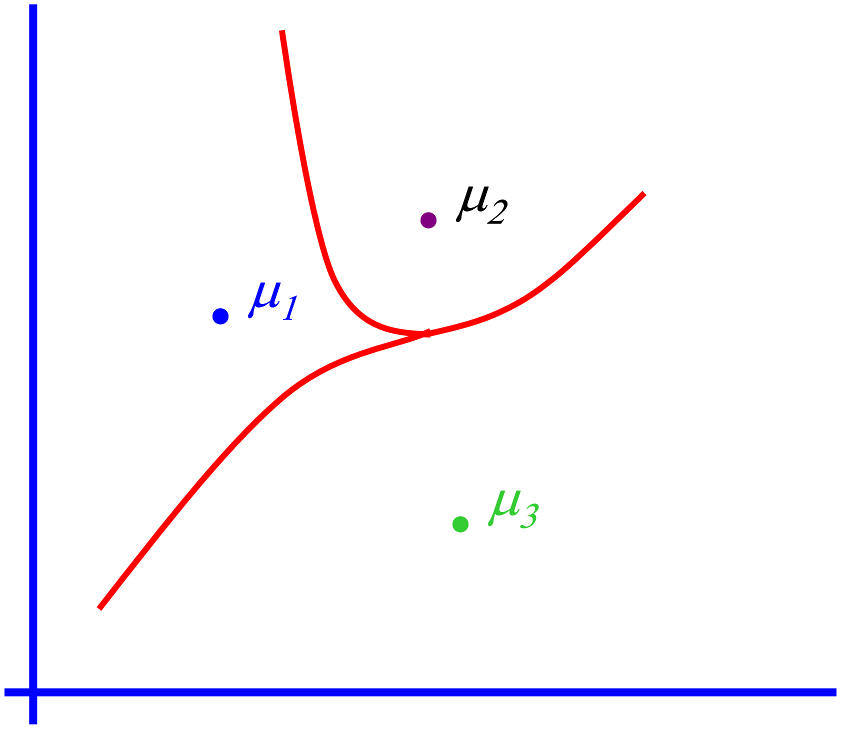
$$p(x) = \sum_i \underset{\substack{\downarrow \\ \textbf{Mixture} \\ \textbf{component}}}{p(x|y=i)} \; \underset{\substack{\downarrow \\ \textbf{Mixture} \\ \textbf{proportion}}}{P(y=i)}$$



General GMM

GMM – Gaussian Mixture Model (Multi-modal distribution)

$$p(x|y=i) \sim \mathrm{N}(\mu_i, \Sigma_i)$$



Gaussian Bayes Classifier:

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

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

$$= \mathrm{x}^{\mathrm{T}} \boxed{\mathrm{W}} \mathrm{x} + \boxed{\mathrm{w}}^{\mathrm{T}} \mathrm{x}$$

→ Depend on $\mu_1, \mu_2, .. , \mu_K, \Sigma_1, \Sigma_2, .. , \Sigma_K$, P(y=1),…, P(Y=k)

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

General GMM

Maximize marginal likelihood:

$$\text{argmax} \prod_j P(x_j) = \text{argmax} \prod_j \sum_{i=1}^{K} P(y_j=i,x_j)$$

$$= \text{argmax} \prod_j \sum_{i=1}^{K} P(y_j=i)p(x_j|y_j=i)$$

Soft assignment: $P(y_j=i) = P(y=i)$

$$= \arg\max \prod_{j=1}^{m} \sum_{i=1}^{k} P(y=i) \frac{1}{\sqrt{\det(\Sigma_i)}} \exp\left[-\frac{1}{2}(x_j-\mu_i)^T \Sigma_i (x_j-\mu_i)\right]$$

How do we find the μ_i's and P(y=i)s which give max. marginal likelihood?

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

* Use gradient descent: Doable, but often slow

Expectation-Maximization (EM)

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

- 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:

  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). Thus it always converges to a local optimum of the likelihood.

Expectation-Maximization (EM)

A simple case:

- We have unlabeled data $\boldsymbol{x}_1\ \boldsymbol{x}_2 \ldots \boldsymbol{x}_m$
- We know there are k classes
- We know P(y=1), P(y=2) P(y=3) … P(y=K)
- We <u>don't</u> know $\boldsymbol{\mu}_1\ \boldsymbol{\mu}_2 \,..\, \boldsymbol{\mu}_k$
- We know common variance σ^2

We can write P(data | $\boldsymbol{\mu}_1 \ldots \boldsymbol{\mu}_k$) $= \mathrm{p}\left(x_1 \ldots x_m \middle| \mu_1 \ldots \mu_k\right)$

$$= \prod_{j=1}^{m} \mathrm{p}\left(x_j \middle| \mu_1 \ldots \mu_k\right)$$

Independent data

$$= \prod_{j=1}^{m} \sum_{i=1}^{k} \mathrm{p}\left(x_j \middle| y=i, \mu_i\right)\ \mathrm{P}(y=i)$$

Marginalize over class

$$\propto \prod_{j=1}^{m} \sum_{i=1}^{k} \exp\left(-\frac{1}{2\sigma^2}\left\|x_j-\mu_i\right\|^2\right)\mathrm{P}(y=i)$$

Expectation (E) step

If we know $\mu_1,\ldots,\mu_k$ $\rightarrow$ easily compute prob. point x_j belongs to class y=i

For each point x_j, j = 1, …, m

$$P\left(y = i \middle| x_j, \mu_1 \ldots \mu_k\right) \propto \exp\left(-\frac{1}{2\sigma^2}\left\|x_j - \mu_i\right\|^2\right) P\left(y = i\right)$$

simply evaluate gaussian and normalize

Equivalent to assigning clusters to each data point in K-means

Maximization (M) step

If we know prob. point x_j belongs to class y=i

→ MLE for μ_i is weighted average

imagine multiple copies of each x_j, each with weight $P(y=i|x_j)$:

$$\mu_i = \frac{\sum_{j=1}^{m} P(y=i|x_j)\, x_j}{\sum_{j=1}^{m} P(y=i|x_j)}$$

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

$$\mathrm{P}(\mathrm{y}=\mathrm{i}|\mathrm{x}_{\mathrm{j}},\mu_1\ldots\mu_{\mathrm{k}}) \propto \exp\left(-\frac{1}{2\sigma^2}\|\mathrm{x}_{\mathrm{j}}-\mu_{\mathrm{i}}\|^2\right)\mathrm{P}(\mathrm{y}=\mathrm{i})$$

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)

$$\mu_i = \frac{\sum_{j=1}^{m} P(y=i|x_j)x_j}{\sum_{j=1}^{m} P(y=i|x_j)}$$

Exactly same as MLE with weighted data

Iterate.

EM for general GMMs

Iterate. On iteration t let our estimates be

$\lambda_t = \{ \mu_1^{(t)}, \mu_2^{(t)} \dots \mu_k^{(t)}, \Sigma_1^{(t)}, \Sigma_2^{(t)} \dots \Sigma_k^{(t)}, p_1^{(t)}, p_2^{(t)} \dots p_k^{(t)} \}$

$p_i^{(t)}$ is shorthand for estimate of *P(y=i)* on t'th iteration

E-step

Compute "expected" classes of all datapoints for each class

$$\mathrm{P}\left(y = i \middle| x_j, \lambda_t\right) \propto p_i^{(t)} \mathrm{p}\left(x_j \middle| \mu_i^{(t)}, \Sigma_i^{(t)}\right)$$

Just evaluate a Gaussian at x_j

M-step

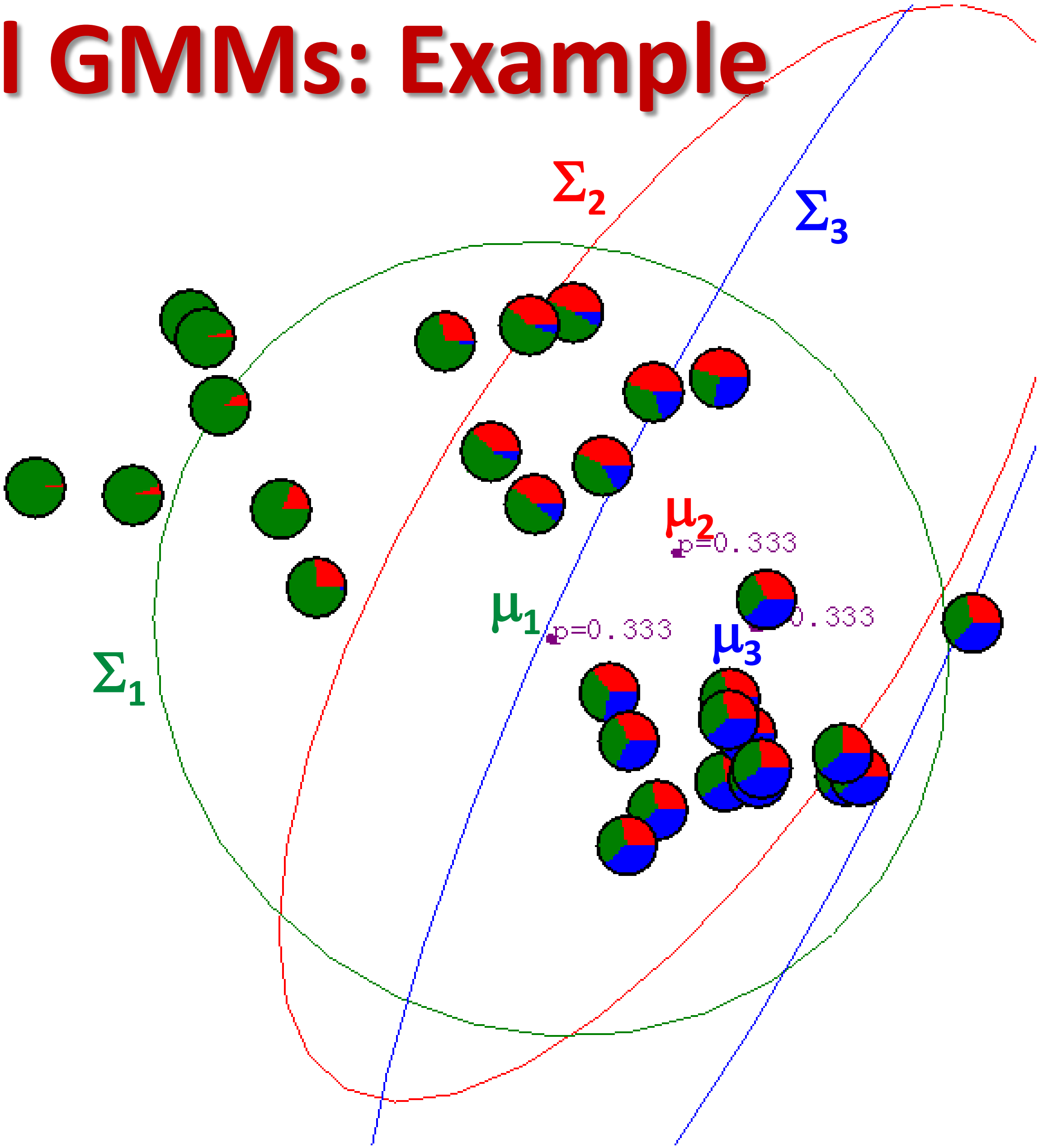
Compute MLEs given our data's class membership distributions (weights)

$$\mu_i^{(t+1)} = \frac{\sum_j \mathrm{P}\left(y = i \middle| x_j, \lambda_t\right) x_j}{\sum_j \mathrm{P}\left(y = i \middle| x_j, \lambda_t\right)} \qquad \Sigma_i^{(t+1)} = \frac{\sum_j \mathrm{P}\left(y = i \middle| x_j, \lambda_t\right)\left(x_j - \mu_i^{(t+1)}\right)\left(x_j - \mu_i^{(t+1)}\right)^T}{\sum_j \mathrm{P}\left(y = i \middle| x_j, \lambda_t\right)}$$

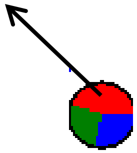
$$p_i^{(t+1)} = \frac{\sum_j \mathrm{P}\left(y = i \middle| x_j, \lambda_t\right)}{m}$$

m = #data points

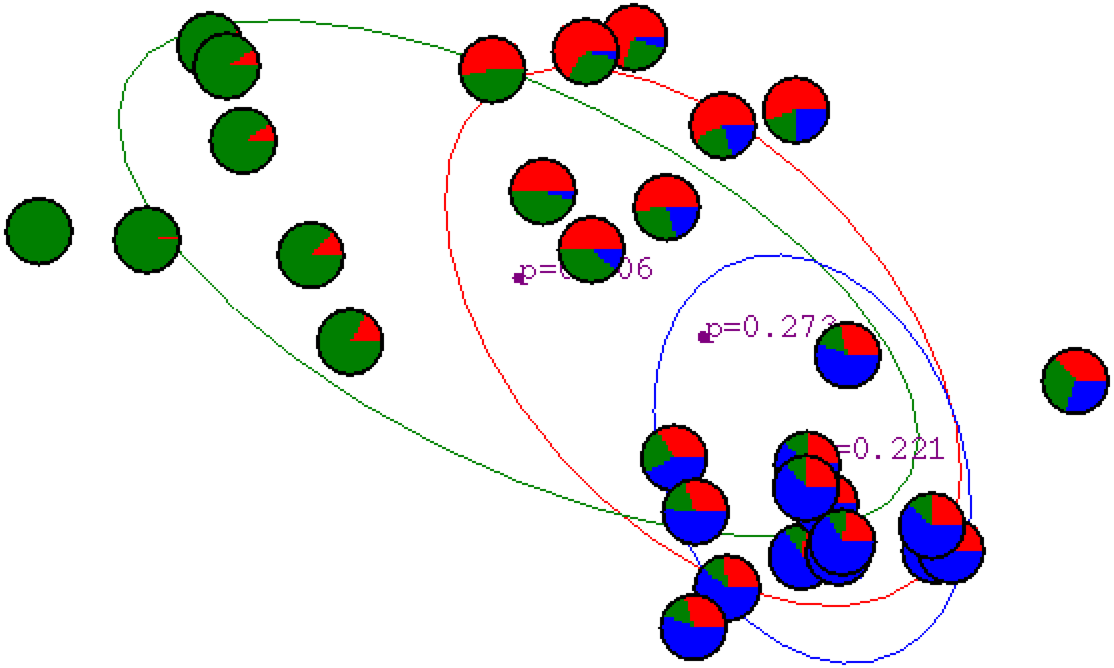
EM for general GMMs: Example



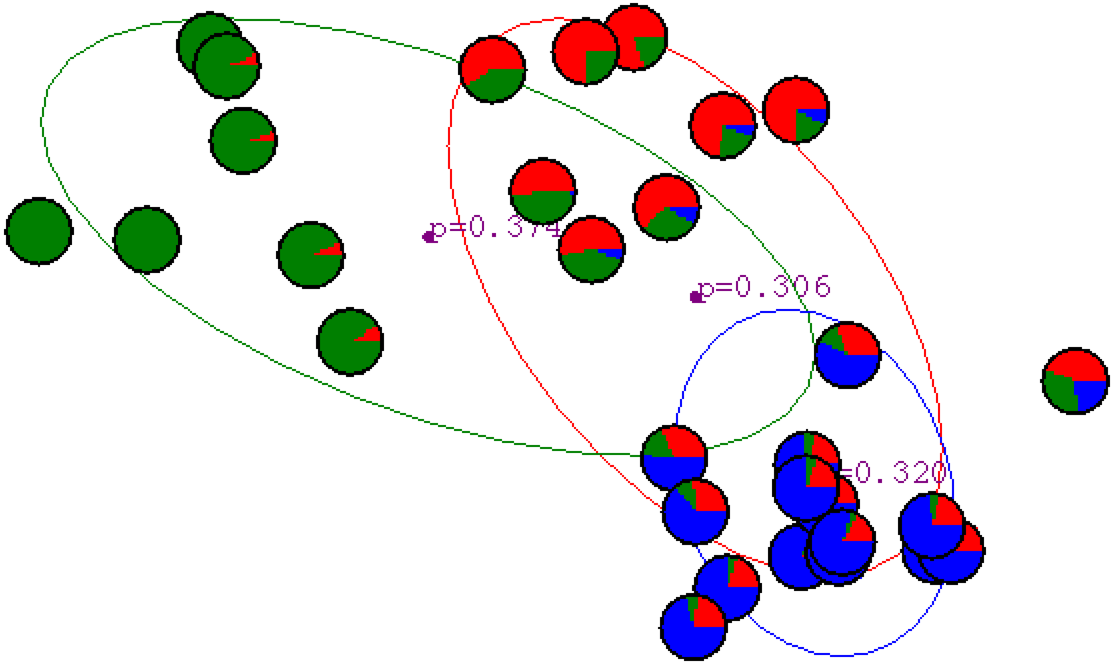
$P(y = ● | x_j, \mu_1, \mu_2, \mu_3, \Sigma_1, \Sigma_2, \Sigma_3, p_1, p_2, p_3)$



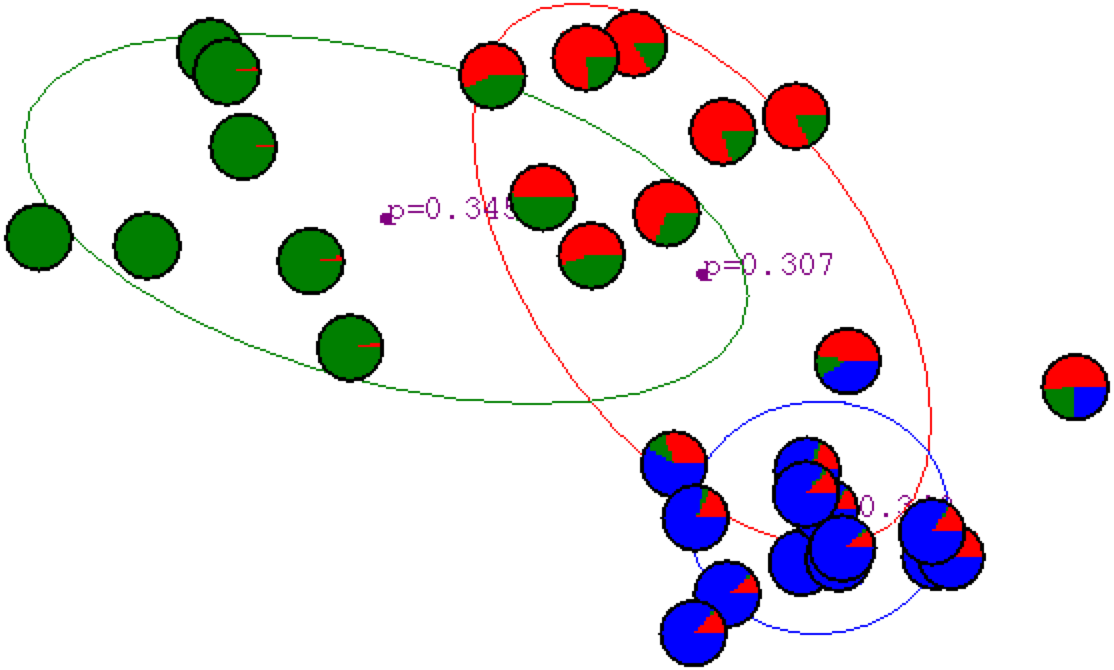
After 1st iteration



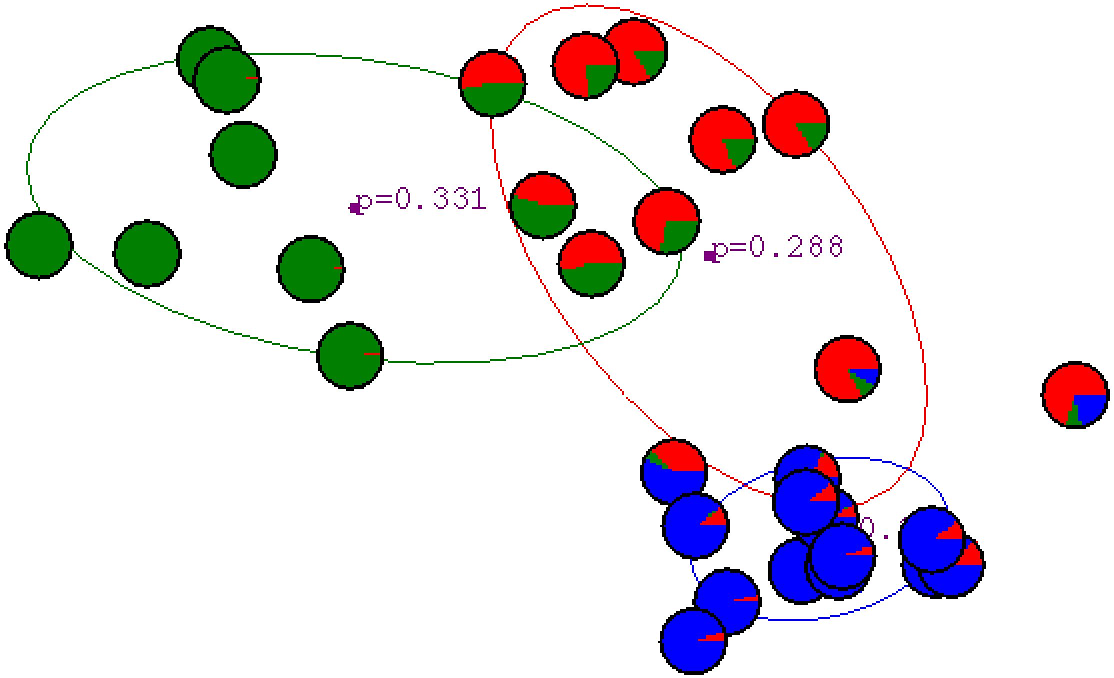
After 2nd iteration



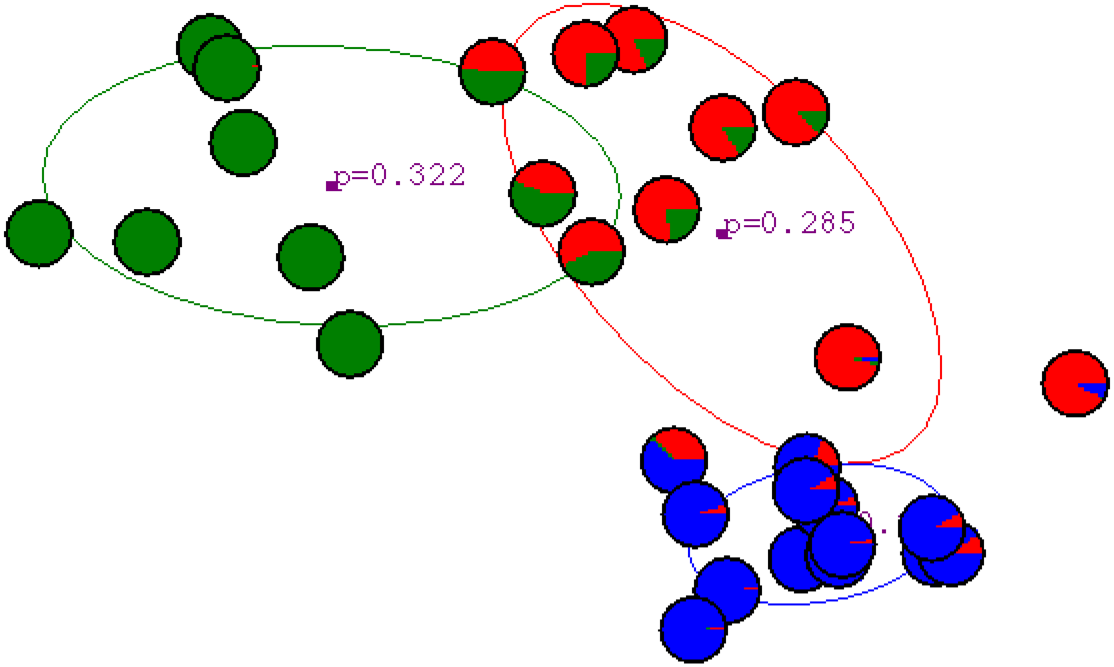
After 3rd iteration



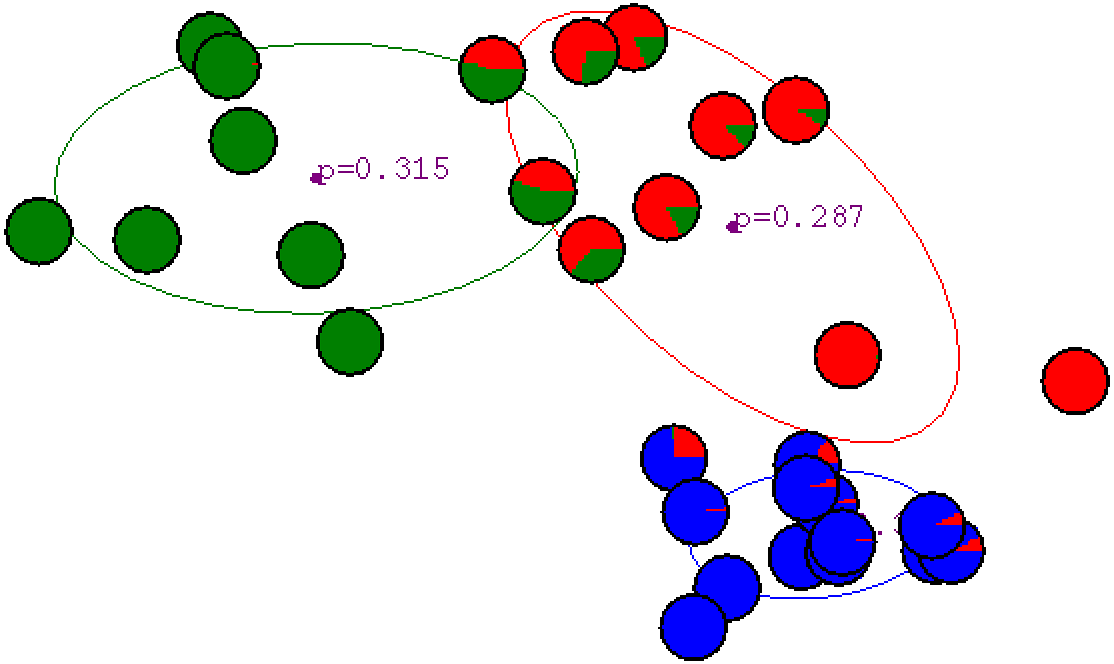
After 4th iteration



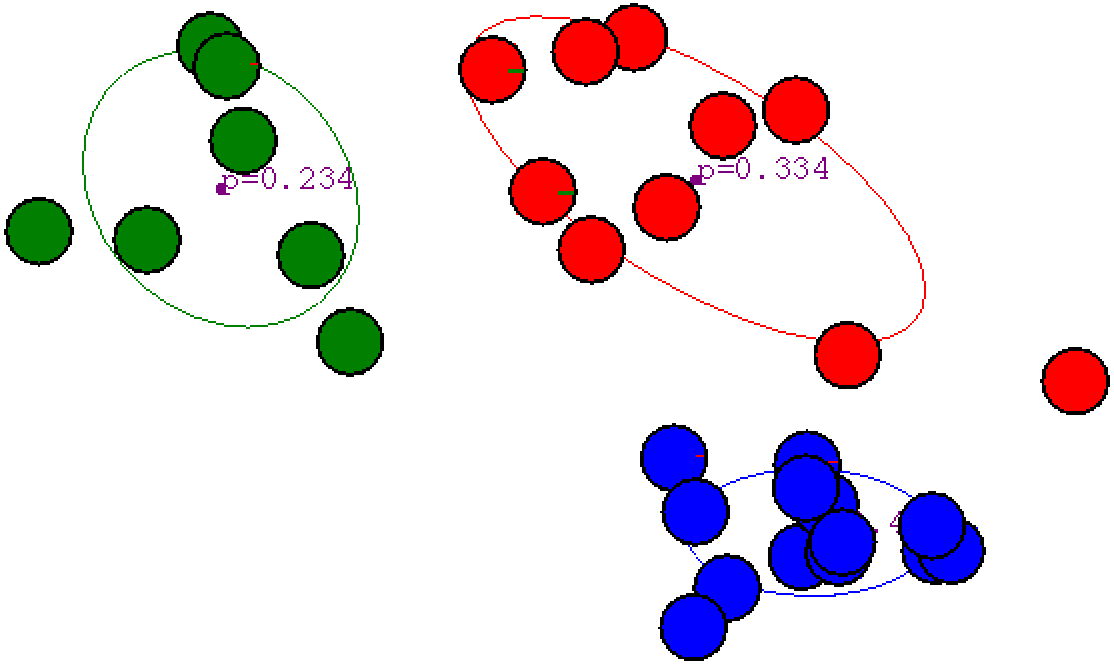
After 5th iteration



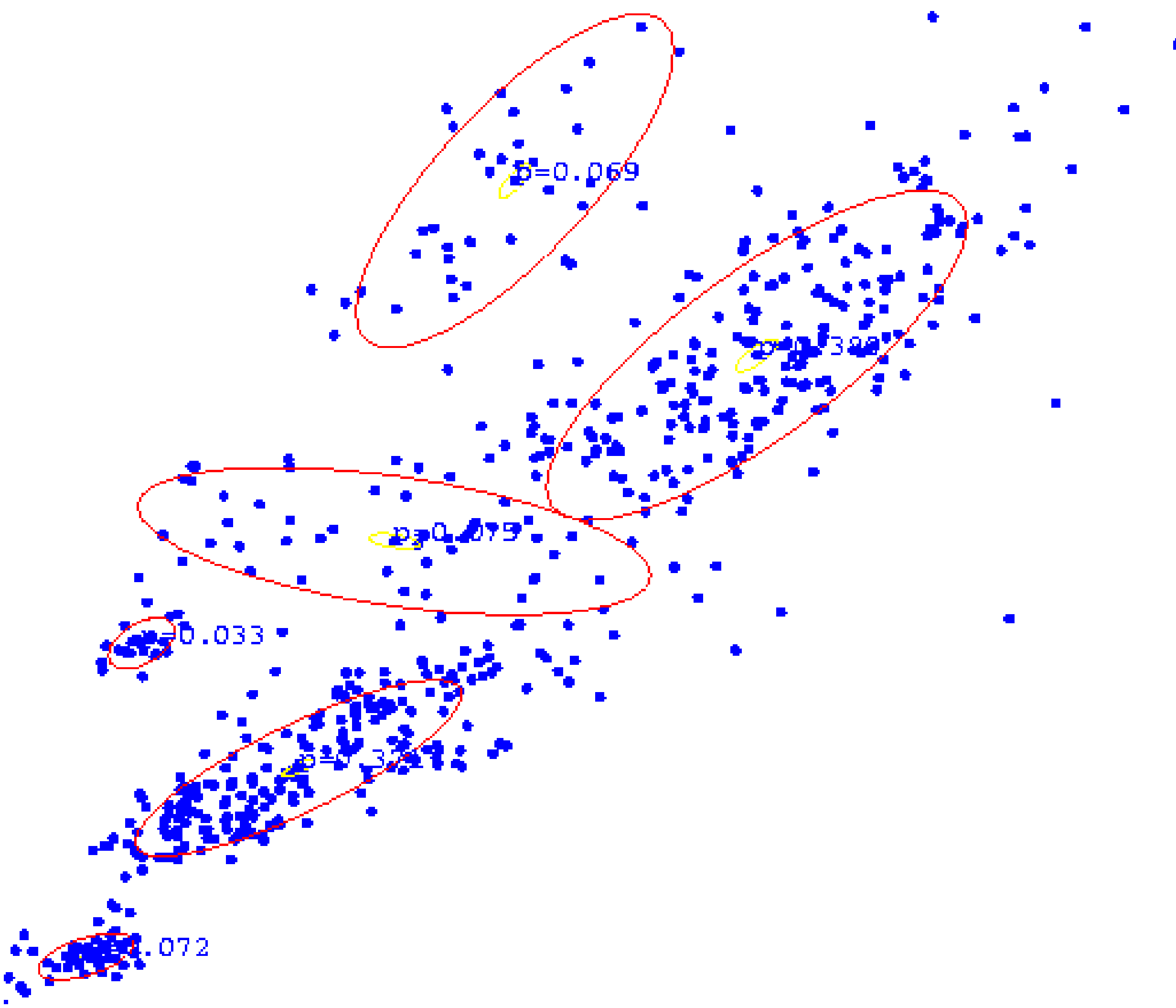
After 6th iteration



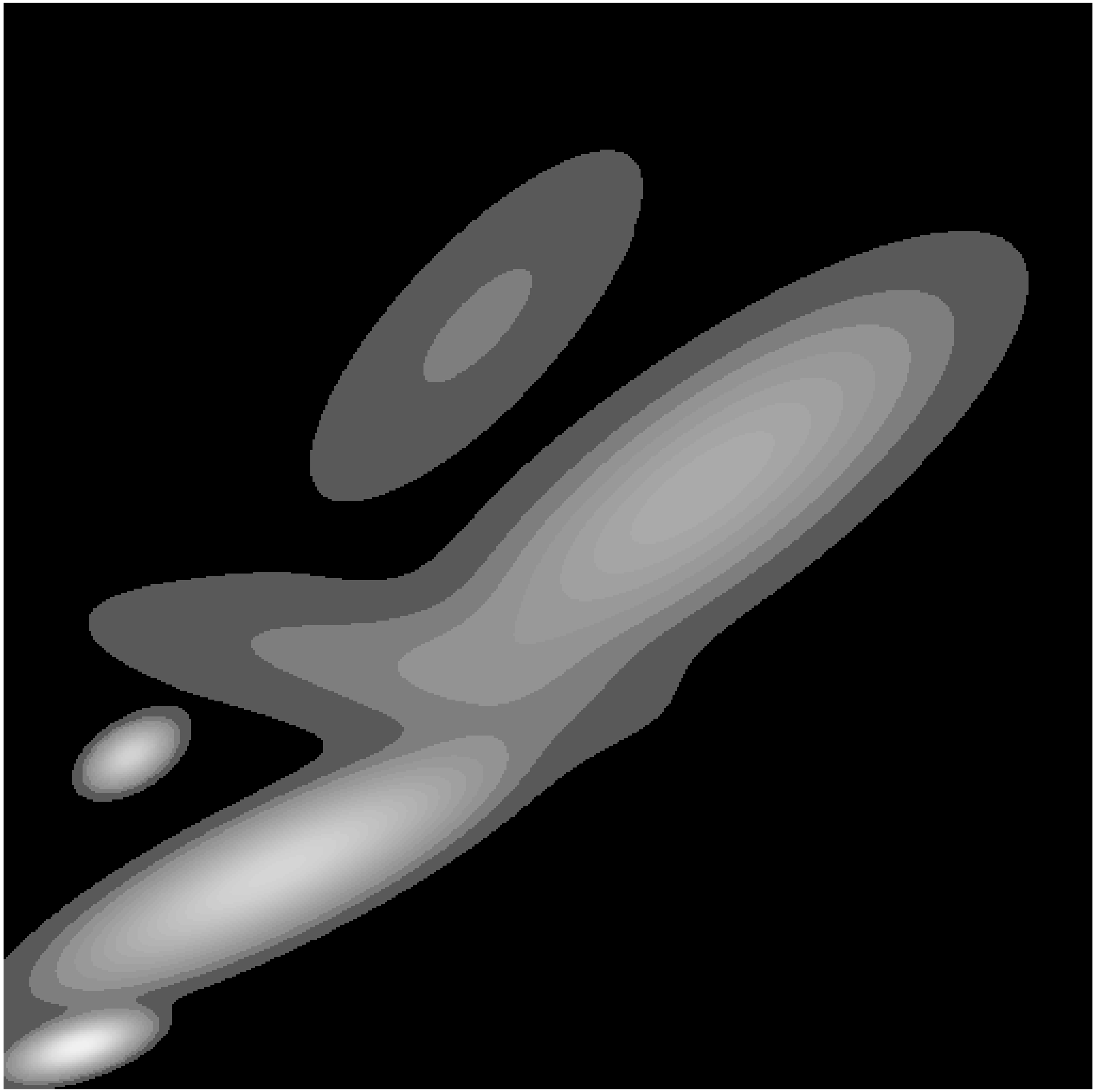
After 20th iteration



GMM clustering of the assay data

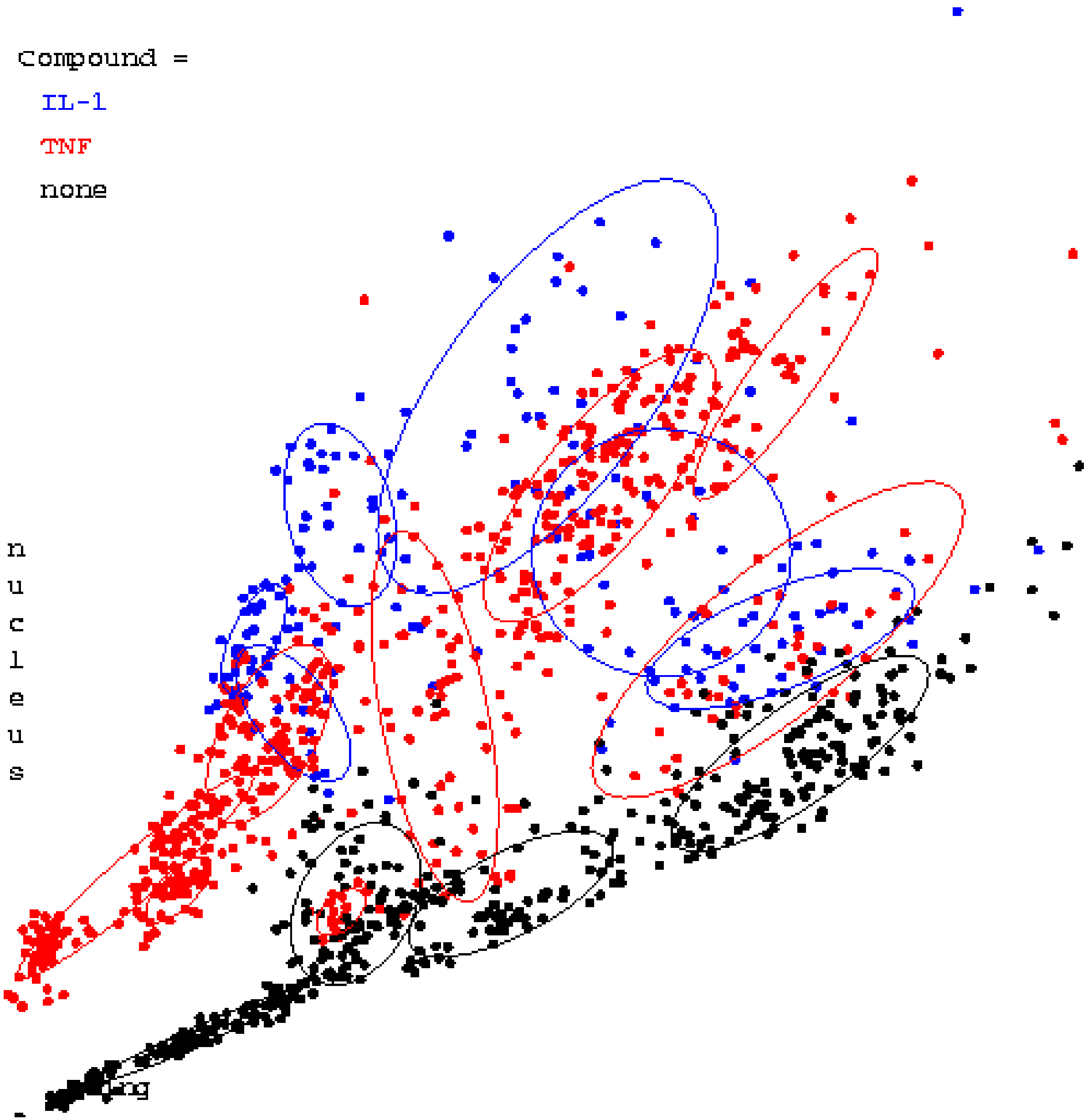


Resulting Density Estimator

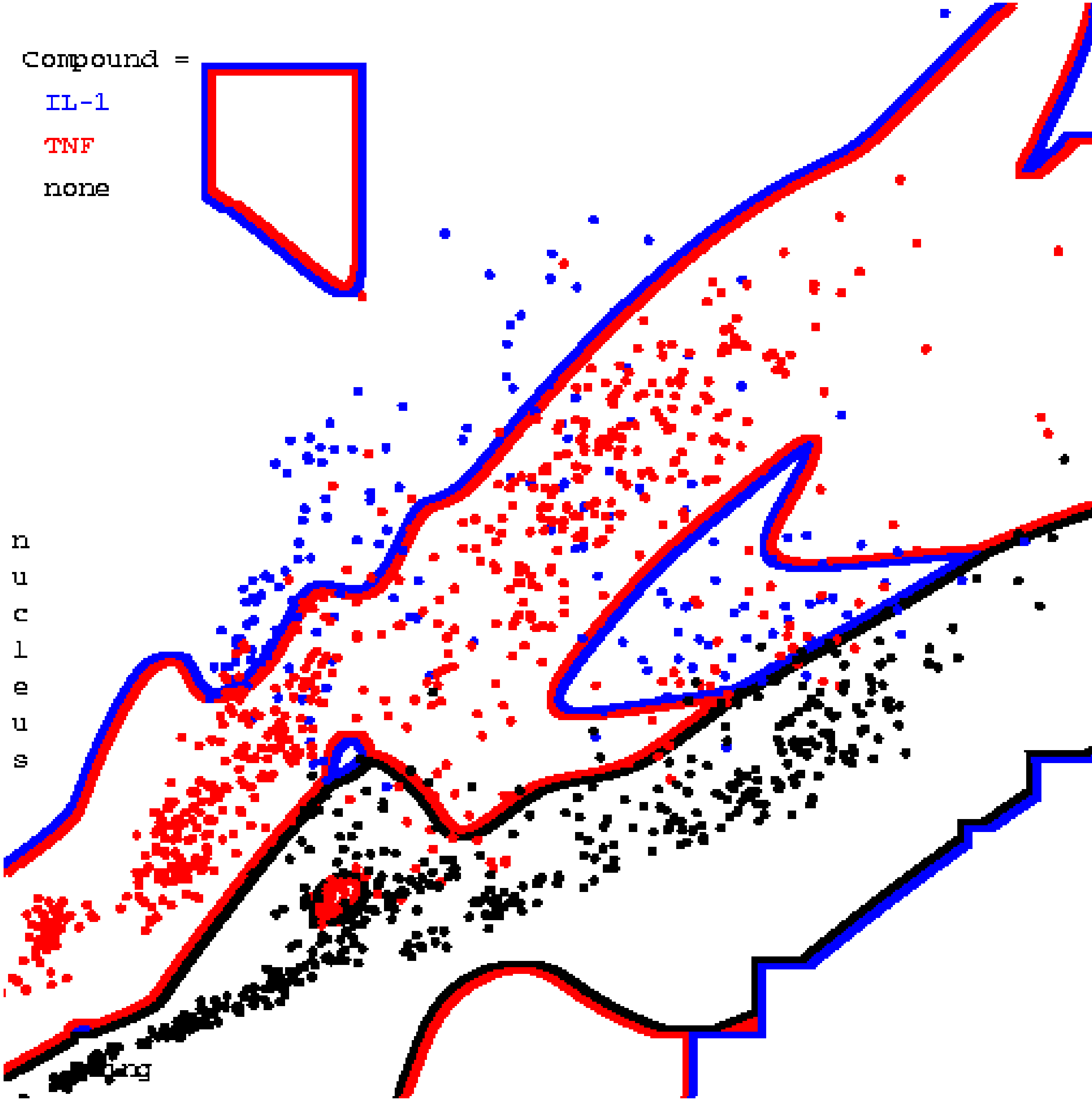


Three classes of assay

(each learned with it’s own mixture model)



Resulting Bayes Classifier



General EM algorithm

Marginal likelihood – **x** is observed, **z** is missing:

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

$$\mathrm{D} = \{\mathrm{x}_j\}_{j=1}^{m}$$

θ - model parameter(s)

E step

x is observed, **z** is missing

Compute probability of missing data given current choice of θ

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

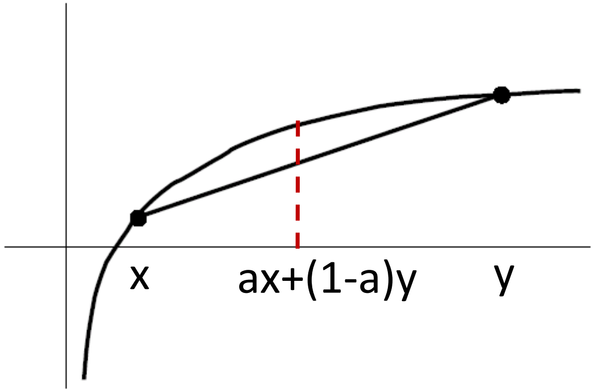
$$\text{E.g., } \mathrm{P}\left(\mathrm{y} = \mathrm{i} \middle| \mathrm{x}_{\mathrm{j}}, \lambda_{\mathrm{t}}\right)$$

M step – Compute estimate of θ by maximizing marginal likelihood using $Q^{(t+1)}(\mathbf{z}|\mathbf{x}_j)$

Lower-bound on marginal likelihood

$$
\begin{aligned}
\mathrm{P}(\mathrm{D};\theta) &= \sum_{j=1}^{m} \log \sum_{\mathbf{z}} P(\mathbf{x}_j, \mathbf{z} \mid \theta) \\
&= \sum_{j=1}^{m} \log \sum_{\mathbf{z}} \underbrace{Q(\mathbf{z} \mid \mathbf{x}_j)}_{\mathrm{P}(\mathbf{z})} \underbrace{\frac{P(\mathbf{z}, \mathbf{x}_j \mid \theta)}{Q(\mathbf{z} \mid \mathbf{x}_j)}}_{\mathrm{f}(\mathbf{z})}
\end{aligned}
$$

Jensen's inequality: $\log \sum_{\mathbf{z}} \mathrm{P}(\mathbf{z})\, \mathrm{f}(\mathbf{z}) \geq \sum_{\mathbf{z}} \mathrm{P}(\mathbf{z}) \log \mathrm{f}(\mathbf{z})$



log: concave function

log(ax+(1-a)y) ≥ a log(x) + (1-a) log(y)

Lower-bound on marginal likelihood

$$\mathrm{P}(\mathrm{D};\theta) = \sum_{j=1}^{m} \log \sum_{\mathbf{z}} P(\mathbf{x}_j, \mathbf{z} \mid \theta)$$

$$= \sum_{j=1}^{m} \log \sum_{\mathbf{z}} \underbrace{Q(\mathbf{z} \mid \mathbf{x}_j)}_{\mathrm{P}(\mathbf{z})} \underbrace{\frac{P(\mathbf{z}, \mathbf{x}_j \mid \theta)}{Q(\mathbf{z} \mid \mathbf{x}_j)}}_{\mathrm{f}(\mathbf{z})}$$

Jensen’s inequality: log $\sum_{\mathbf{z}}$ P(**z**) f(**z**) ≥ $\sum_{\mathbf{z}}$ P(**z**) log f(**z**)

$$\geq \sum_{j=1}^{m} \sum_{\mathbf{z}} Q(\mathbf{z} \mid \mathbf{x}_j) \log \frac{P(\mathbf{z}, \mathbf{x}_j \mid \theta)}{Q(\mathbf{z} \mid \mathbf{x}_j)}$$

$$= \sum_{j=1}^{m} \sum_{\mathbf{z}} Q(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta) + \underbrace{m.H(Q)}_{\text{Independent of } \theta}$$

Lower-bound on marginal likelihood

$$\mathrm{P}(\mathrm{D};\theta) \;\geq\; \sum_{j=1}^{m}\sum_{\mathbf{z}} Q(\mathbf{z}\mid\mathbf{x}_j)\log P(\mathbf{z},\mathbf{x}_j\mid\theta) + \underbrace{m.H(Q)}_{\text{Independent of }\theta}$$

$$\parallel$$

$$\underbrace{\sum_{\mathbf{z}}\sum_{j=1}^{m}\log P(\mathbf{z},\mathbf{x}_j\mid\theta)Q(\mathbf{z}\mid\mathbf{x}_j)}_{\text{Expected log likelihood wrt Q}}$$

Since **z** is missing, instead take expectation over it (recall: probability of missing data **z** computed in E-step)

M step

$$\mathrm{P}(\mathrm{D};\theta) \geq \sum_{\mathbf{z}} \sum_{j=1}^{m} \log P(\mathbf{z}, \mathbf{x}_j \mid \theta) Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) + m.H(Q)$$

Maximize lower bound on marginal likelihood

$$\theta^{(t+1)} \leftarrow \arg\max_{\theta} \underbrace{\sum_{\mathbf{z}} \sum_{j=1}^{m} \log P(\mathbf{z}, \mathbf{x}_j \mid \theta) Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j)}_{\text{Expected log likelihood wrt } Q^{(t+1)}}$$

Use expected counts instead of counts when computing MLE:

If learning requires Count(**x**,**z**), Use $E_{Q(t+1)}$[Count(**x**,**z**)]

EM as Coordinate Ascent

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

M-step: Fix Q, maximize F over θ

$$\mathrm{P}(\mathrm{D};\theta) \;\geq\; F(\theta, Q^{(t)}) = \sum_{j=1}^{m} \sum_{\mathbf{z}} Q^{(t)}(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta) + m.H(Q^{(t)})$$

M-step maximizes lower bound F on marginal likelihood => doesn't decrease the marginal likelihood

E-step: Fix θ, maximize F over Q

We'll show this next. Thus,

E-step also maximizes lower bound F on marginal likelihood => doesn't decrease the marginal likelihood

Since marginal likelihood is bounded, Convergence follows!

Convergence of EM

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

E-step: Fix θ, maximize F over Q

$$\mathrm{P}(\mathrm{D};\theta^{(t)}) \geq \; F(\theta^{(t)}, Q) = \sum_{j=1}^{m} \sum_{\mathbf{z}} Q(\mathbf{z} \mid \mathbf{x}_j) \log \frac{P(\mathbf{z}, \mathbf{x}_j \mid \theta^{(t)})}{Q(\mathbf{z} \mid \mathbf{x}_j)}$$

$$= \sum_{j=1}^{m} \sum_{\mathbf{z}} Q(\mathbf{z} \mid \mathbf{x}_j) \log \frac{P(\mathbf{z} \mid \mathbf{x}_j, \theta^{(t)}) \, P(\mathbf{x}_j|\theta^{(t)})}{Q(\mathbf{z} \mid \mathbf{x}_j)}$$

$$= \underbrace{\sum_{j=1}^{m} \sum_{\mathbf{z}} Q(\mathbf{z} \mid \mathbf{x}_j) \log \frac{P(\mathbf{z} \mid \mathbf{x}_j, \theta^{(t)})}{Q(\mathbf{z} \mid \mathbf{x}_j)}}_{-KL(Q(\mathbf{z}|\mathbf{x}_j),\, P(\mathbf{z}|\mathbf{x}_j, \theta^{(t)}))} + \underbrace{\sum_{j=1}^{m} \cancelto{1}{\sum_{\mathbf{z}} Q(\mathbf{z}|\mathbf{x}_j)} \log P(\mathbf{x}_j|\theta^{(t)})}_{\mathrm{P}(\mathrm{D};\theta^{(t)})}$$

KL divergence between two distributions

Convergence of EM

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

E-step: Fix θ, maximize F over Q

$$\mathrm{P}(\mathrm{D};\theta^{(t)}) \geq \;\; F(\theta^{(t)}, Q) = \sum_{j=1}^{m} \sum_{\mathbf{z}} Q(\mathbf{z} \mid \mathbf{x}_j) \log \frac{P(\mathbf{z}, \mathbf{x}_j \mid \theta^{(t)})}{Q(\mathbf{z} \mid \mathbf{x}_j)}$$

$$= \sum_{j=1}^{m} -KL(Q(\mathbf{z}|\mathbf{x}_j), P(\mathbf{z}|\mathbf{x}_j, \theta^{(t)})) \; + \; \mathrm{P}(\mathrm{D};\theta^{(t)})$$

KL>=0, Maximized if KL divergence = 0 **KL(Q,P) = 0 iif Q = P**

Recall E-step: $Q^{(t+1)}(\mathbf{z} \mid \mathbf{x}_j) \;=\; P(\mathbf{z} \mid \mathbf{x}_j, \theta^{(t)})$

Thus, E-step also maximizes lower bound F on marginal likelihood => doesn't decrease the marginal likelihood

Convergence of EM

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

M-step: Fix Q, maximize F over θ

$$\mathrm{P}(\mathrm{D};\theta) \;\geq\; F(\theta, Q^{(t)}) = \sum_{j=1}^{m} \sum_{\mathbf{z}} Q^{(t)}(\mathbf{z} \mid \mathbf{x}_j) \log P(\mathbf{z}, \mathbf{x}_j \mid \theta) + m.H(Q^{(t)})$$

Maximizes lower bound F on marginal likelihood

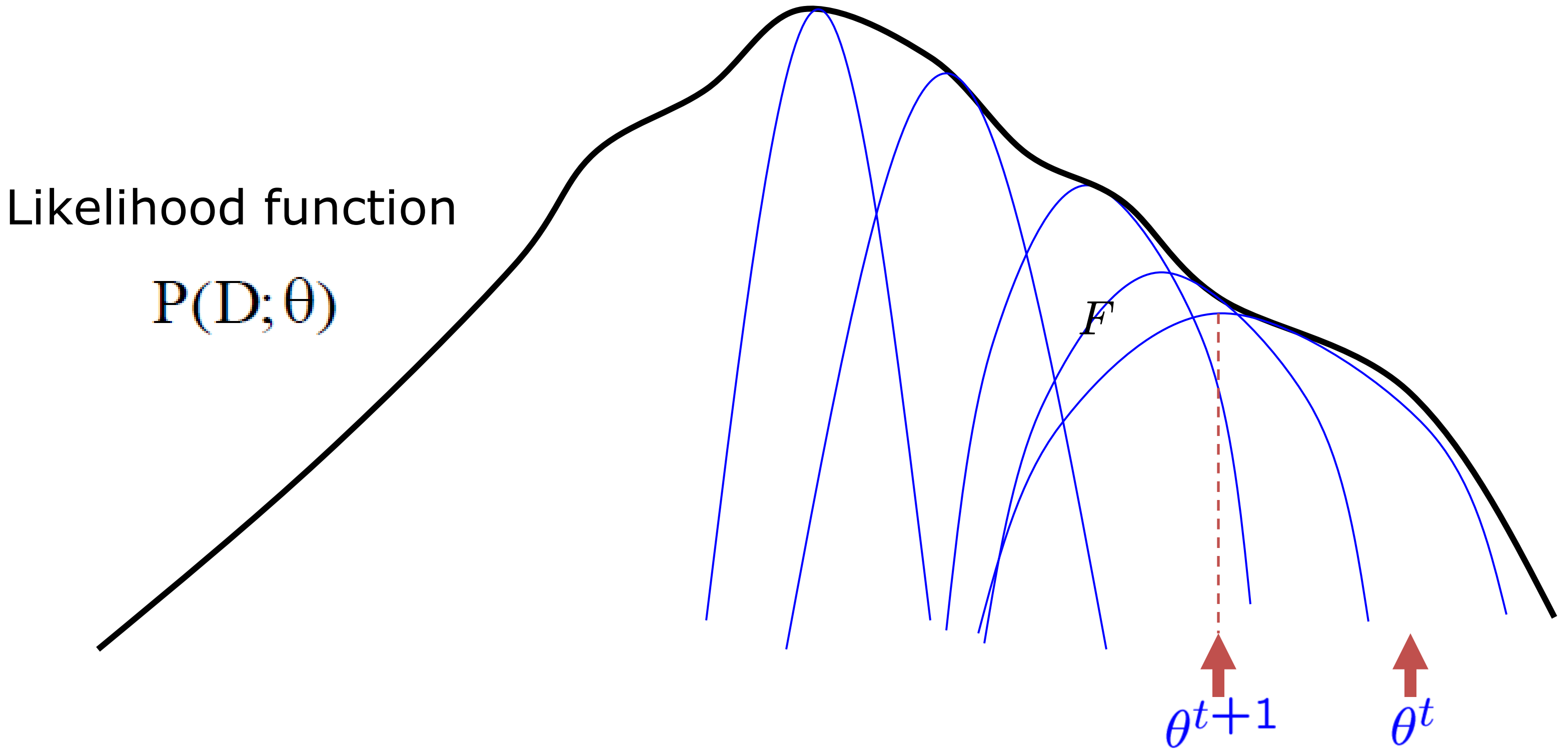
E-step: Fix θ, maximize F over Q

$$\mathrm{P}(\mathrm{D};\theta^{(t)}) \geq F(\theta^{(t)}, Q) \;= \mathrm{P}(\mathrm{D};\theta^{(t)}) - \sum_{j=1}^{m} KL\left(Q(\mathbf{z} \mid \mathbf{x}_j) || P(\mathbf{z} \mid \mathbf{x}_j, \theta^{(t)})\right)$$

Re-aligns F with marginal likelihood

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

Convergence of EM

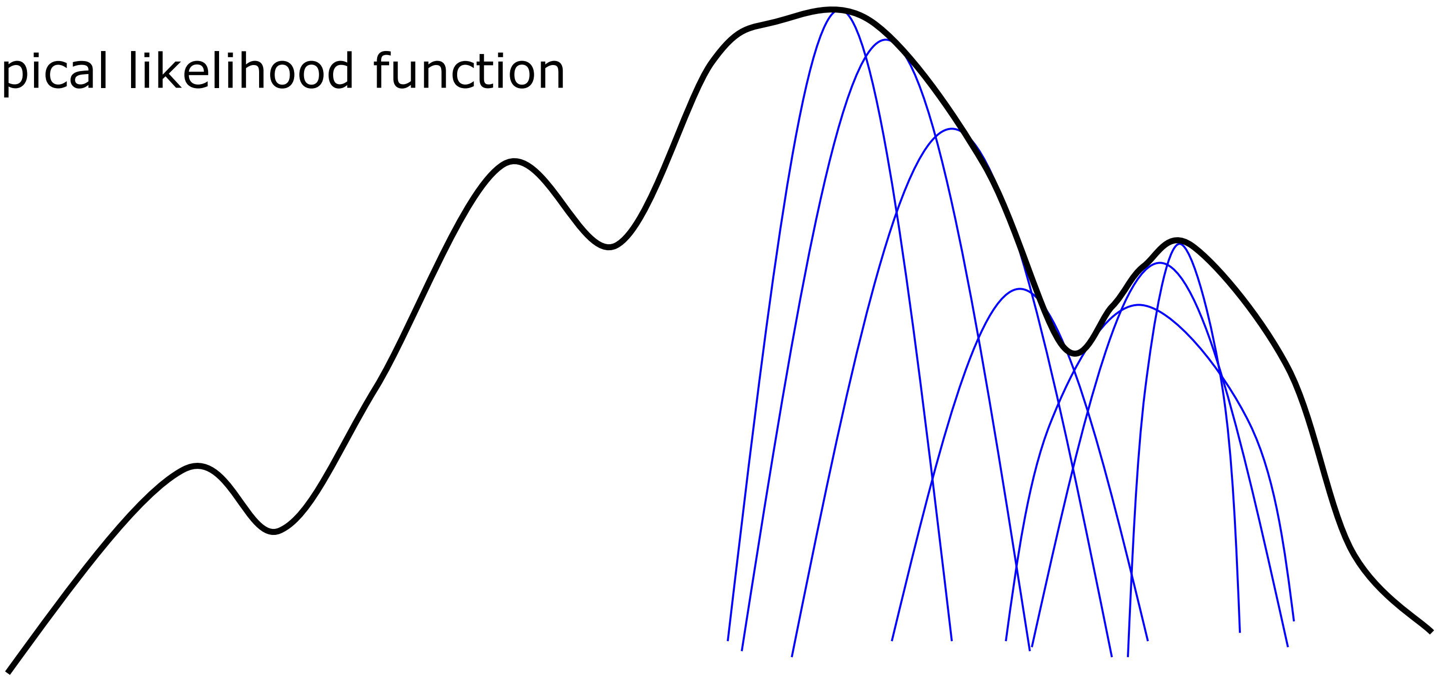


Sequence of EM lower bound F-functions

EM monotonically converges to a local maximum of likelihood !

Convergence of EM

Typical likelihood function



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

Use multiple, randomized initializations in practice

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 maximum likelihood parameter estimates.

- Alternate between filling in the latent variables using the best guess (posterior) and updating the parameters based on this guess:
  1. E-step: $Q^{t+1} = \arg\max_{Q} F(\theta^{t}, Q)$
  2. M-step: $\theta^{t+1} = \arg\max_{\theta} F(\theta, Q^{t+1})$

- In the M-step we optimize a lower bound on the likelihood. In the E-step we close the gap, making bound=likelihood.

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

- BUT Extremely popular in practice.