# EM Algorithm

Aarti Singh

Slides courtesy: Eric Xing, Carlos Guetrin

Machine Learning 10-701/15-781
Feb 17, 2010

ML MACHINE LEARNING DEPARTMENT

Carnegie Mellon.
School of Computer Science

# 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:
  - ☐ $C^{(t)}(j) \leftarrow \arg\min_i ||\mu_i - x_j||^2$
- **Recenter**: $\mu_i$ becomes centroid of its point:
  - ☐ $\mu_i^{(t+1)} \leftarrow \arg\min_\mu \sum_{j:C(j)=i} ||\mu - x_j||^2$
  - ☐ 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:

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

- 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 $\mu$, 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:

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

- **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 ascent on F)

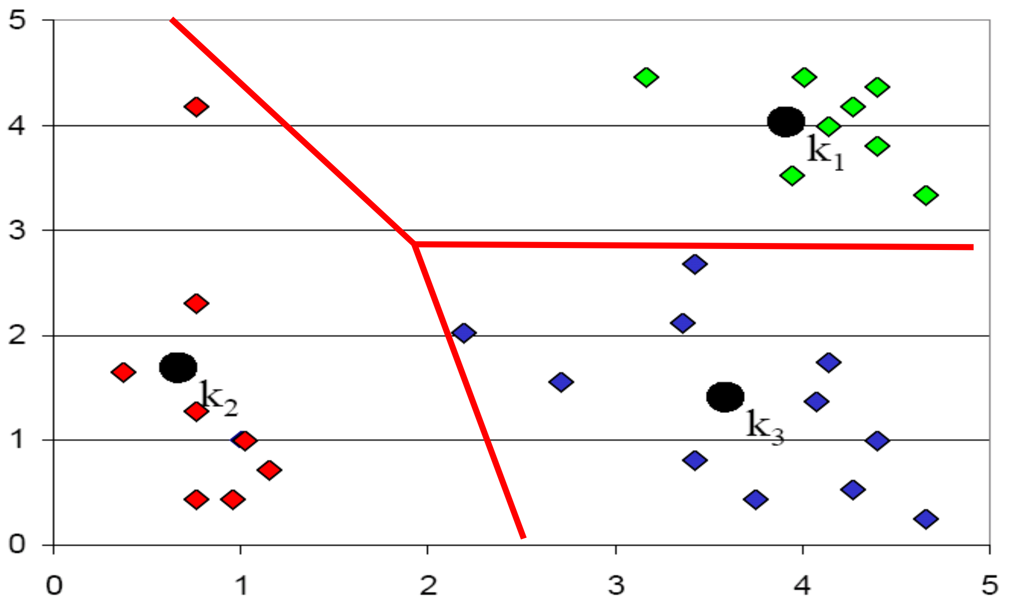
**(1)** Fix $\mu$, optimize C **Expectation step**

**(2)** Fix C, optimize $\mu$ **Maximization step**

Today, we will see a generalization of this approach:

**EM algorithm**

# K-means Decision boundaries



**"Linear" Decision Boundaries**

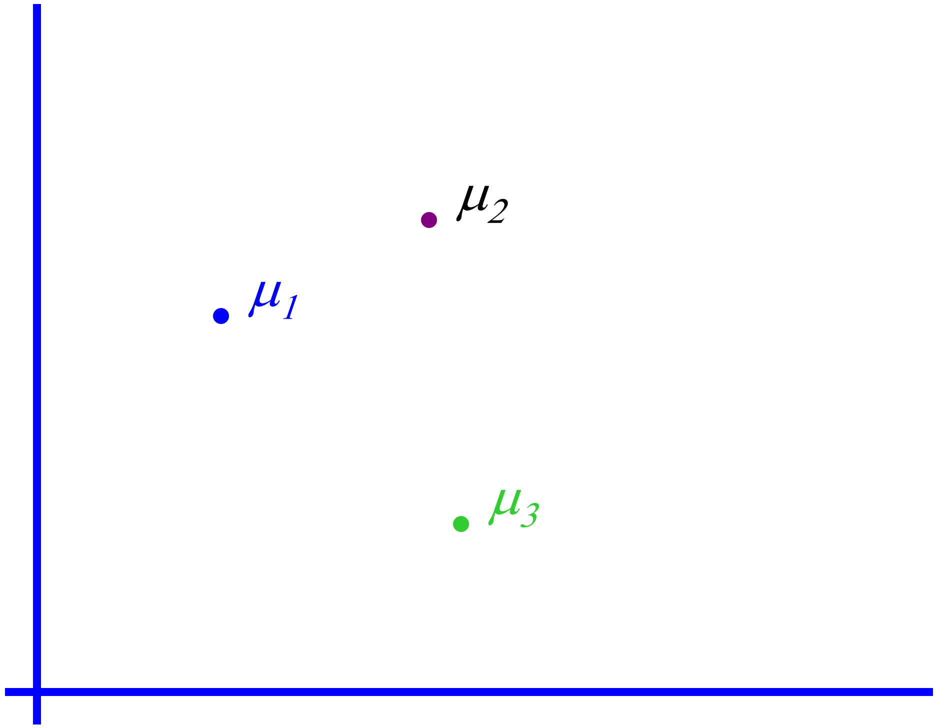
## Generative Model:

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

# K-means: Generative model

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

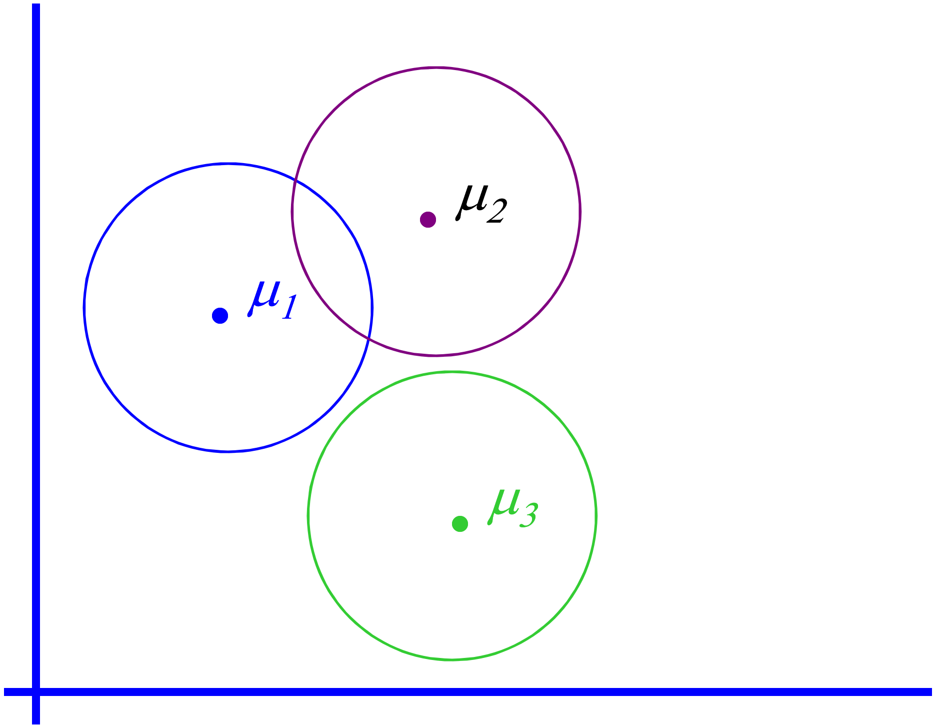
- There are k components
- Component *i* has an associated mean vector $\mu_i$



# K-means: Generative model

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

- There are k components
- Component *i* has an associated mean vector $\mu_i$
- Each component generates data from a Gaussian with mean $\mu_i$ and covariance matrix $\sigma^2 I$



Each data point is generated according to the following recipe:

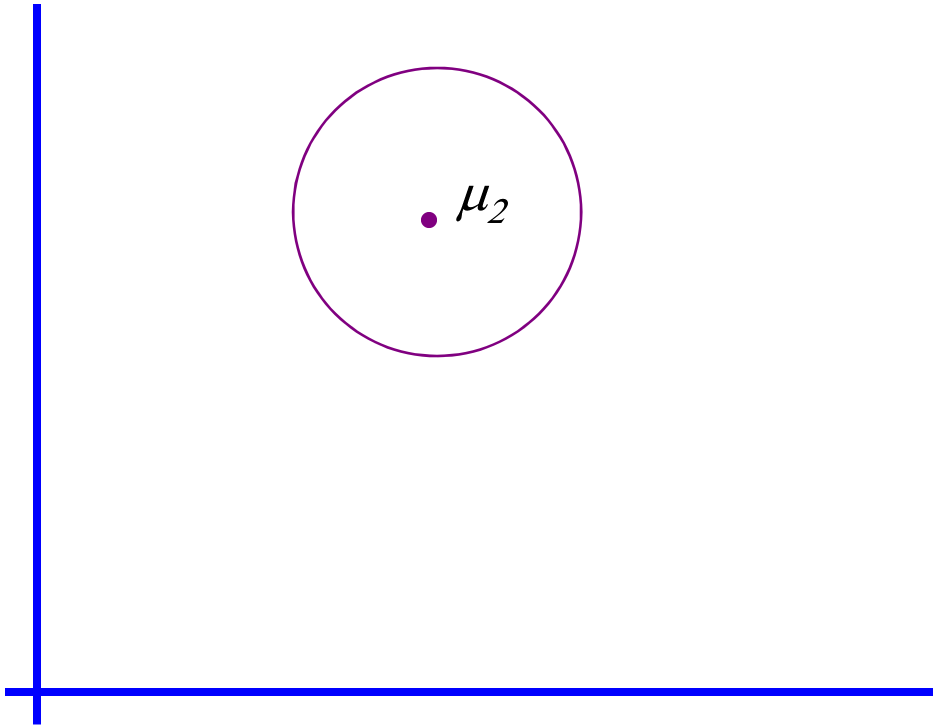
# K-means: Generative model

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

- There are k components
- Component *i* has an associated mean vector $\mu_i$
- Each component generates data from a Gaussian with mean $\mu_i$ and covariance matrix $\sigma^2 I$

Each data point is generated according to the following recipe:

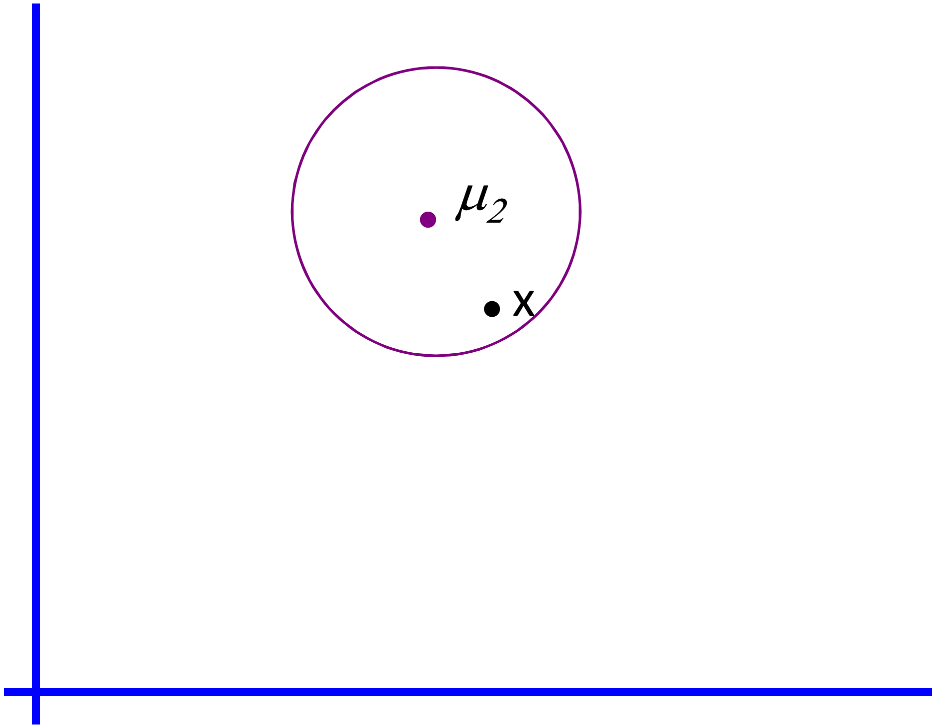
1) Pick a component at random: Choose component i with probability *P(y=i)*



# K-means: Generative model

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

- There are k components
- Component *i* has an associated mean vector $\mu_i$
- Each component generates data from a Gaussian with mean $\mu_i$ and covariance matrix $\sigma^2 I$



Each data point is generated according to the following recipe:

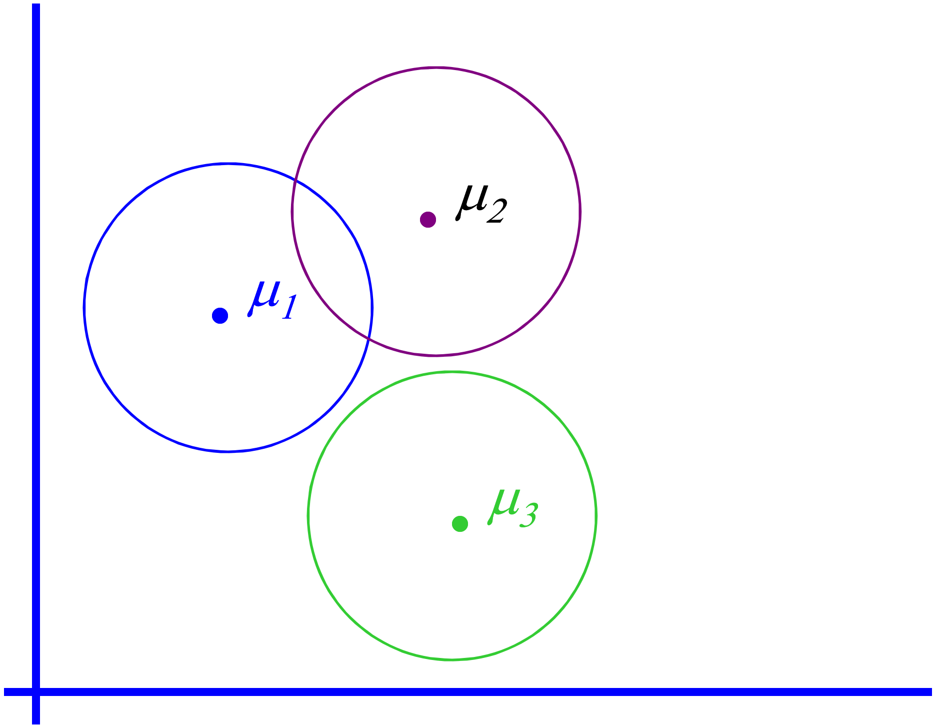
1) Pick a component at random: Choose component i with probability *P(y=i)*
2) Datapoint x ~ $\mathrm{N}(\mu_i, \sigma^2 I)$

# K-means: Generative model

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

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

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



# K-means: Generative model

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

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

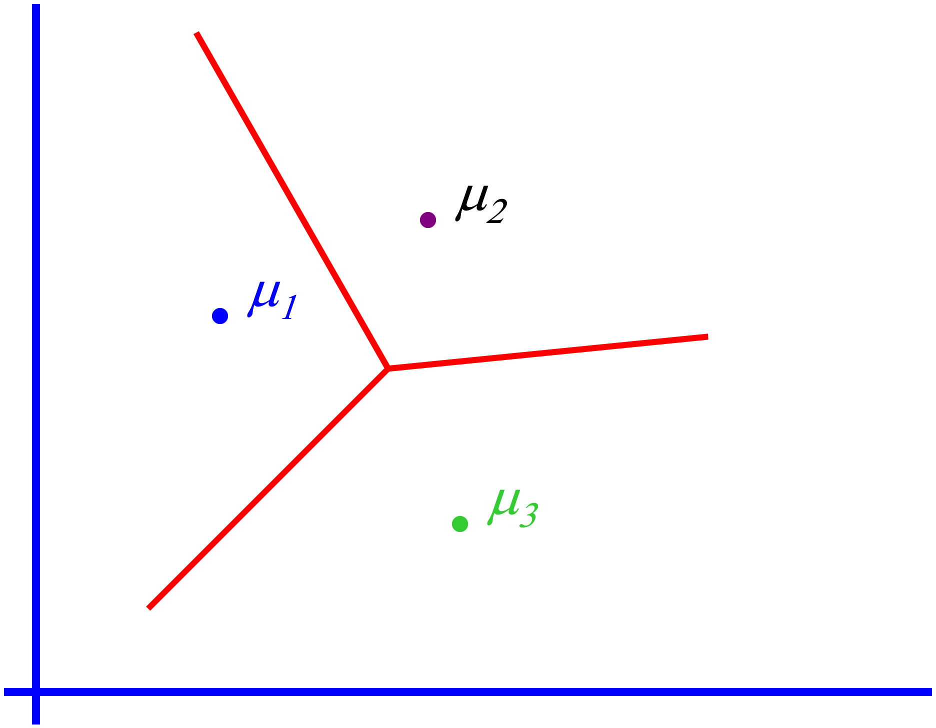
Gaussian Bayes Classifier:

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

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

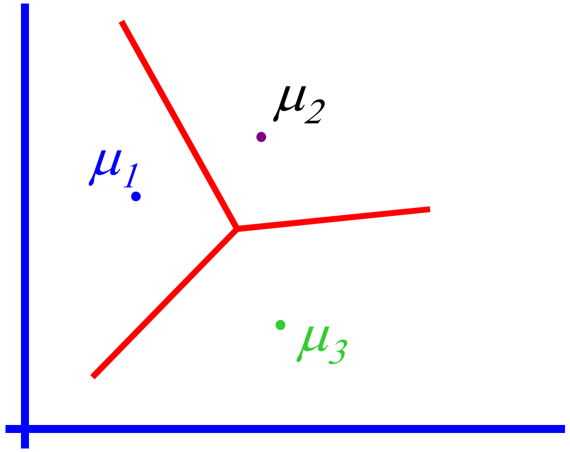
$$= \boxed{w}^{T} x$$

→ 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

# K-means: Generative model



Maximum Likelihood Estimate (MLE)

argmax $\prod_i P(y_i,x_i)$

$\mu_1, \mu_2, .. , \mu_K, \sigma^2,$

P(y=1),…, P(Y=k)

But we don't know $y_i$'s!!!

Maximize marginal likelihood:

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

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

# K-means: Generative model

Maximize marginal likelihood:

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

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

$$P(y_j = i, x_j) \propto P(y_j = i) \exp\left[-\frac{1}{2\sigma^2}\left\|x_j - \mu_i\right\|^2\right]$$

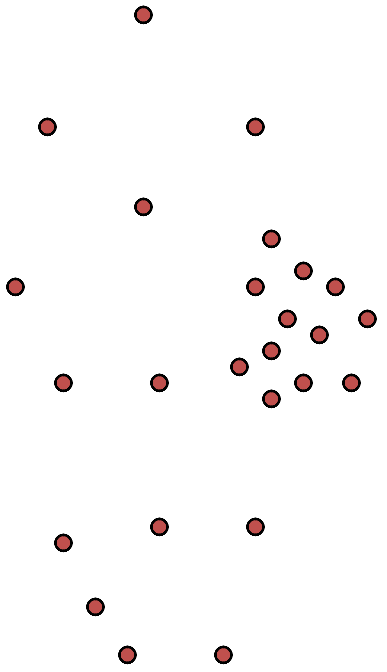
If each $x_j$ belongs to one class C(j) (hard assignment), marginal likelihood:

$P(y_j=i) = 1$ or $0$ $\qquad$ $1$ if $i = C(j)$

$$\prod_{j=1}^{m}\sum_{i=1}^{k} P(y_j = i, x_j) \propto \prod_{j=1}^{m} \exp\left[-\frac{1}{2\sigma^2}\left\|x_j - \mu_{C(j)}\right\|^2\right] = \sum_{j=1}^{m} -\frac{1}{2\sigma^2}\left\|x_j - \mu_{C(j)}\right\|^2$$

**Same as K-means!!!**

# (One) bad case for K-means

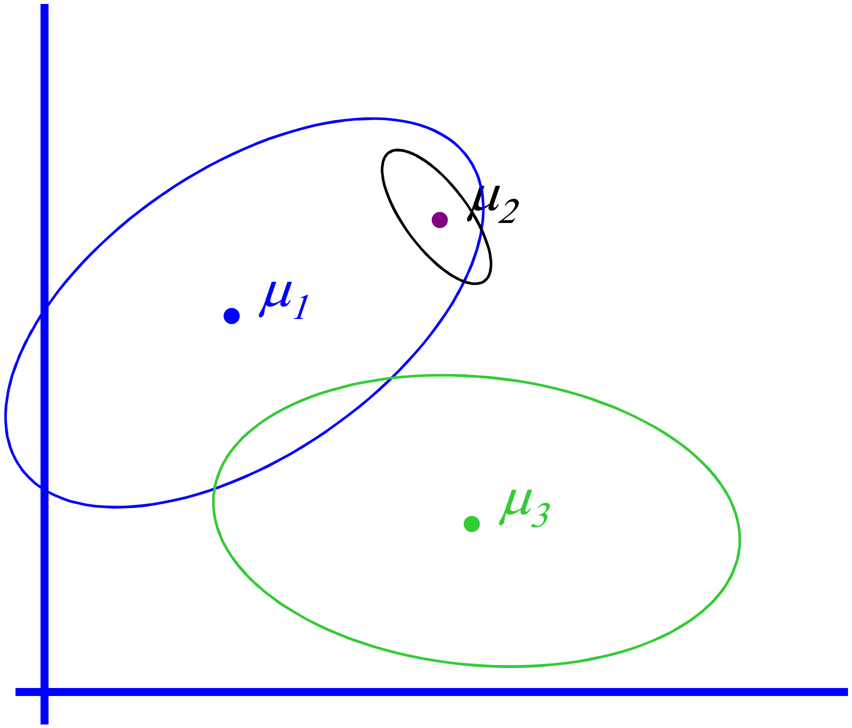


- Clusters may not be linearly separable
- Clusters may overlap
- Some clusters may be “wider” than others

# General GMM

GMM – Gaussian Mixture Model (Multi-modal distribution)

- There are k components
- Component *i* has an associated mean vector $\mu_i$
- Each component generates data from a Gaussian with mean $\mu_i$ and covariance matrix $\Sigma_i$



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 ~ $N(\mu_i, \Sigma_i)$

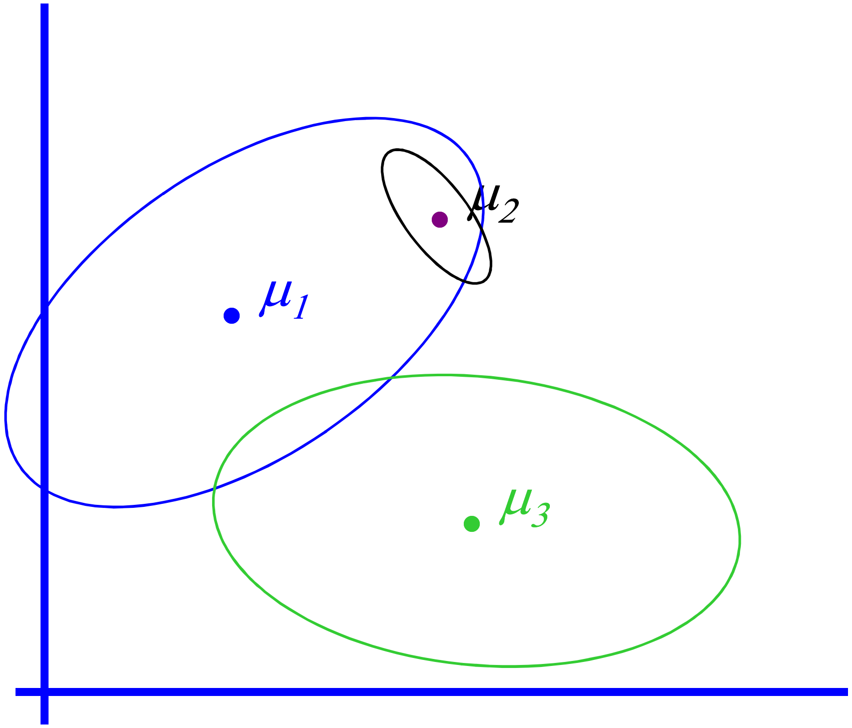
# General GMM

GMM – Gaussian Mixture Model (Multi-modal distribution)

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

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

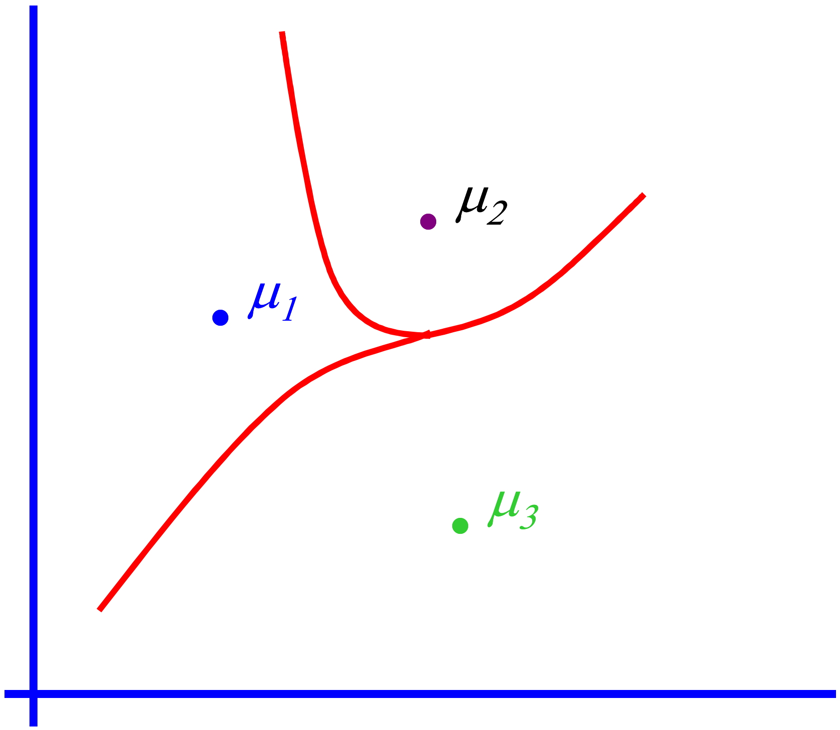
**Mixture component** **Mixture proportion**



# General GMM

GMM – Gaussian Mixture Model (Multi-modal distribution)

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



Gaussian Bayes Classifier:

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

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

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

→ 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:

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

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

Uncertain about class of each $x_j$ (soft assignment), $P(y_j=i) = P(y=i)$

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

How do we find the $\mu_i$'s which give max. marginal likelihood?

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

* Use gradient descent: Often slow but doable

# 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.
  - Enforces constraints automatically.
  - Calls inference and fully observed learning as subroutines.
- 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 don't know $\boldsymbol{\mu}_1\ \boldsymbol{\mu}_2\ ..\ \boldsymbol{\mu}_k$
We know common variance $\sigma^2$

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

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

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 $\mu_1,\ldots,\mu_k$ → easily compute prob. point $x_j$ belongs to class y=i

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

Simply evaluate gaussian and normalize

# Maximization (M) step

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

→ MLE for $\mu_i$ is weighted average

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

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

# EM for spherical, same variance GMMs

**E-step**

Compute "expected" classes of all datapoints for each class

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

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

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

# EM for axis-aligned GMMs

$$\Sigma_i = \begin{pmatrix} \sigma^2{}_{i,1} & 0 & 0 & \cdots & 0 & 0 \\ 0 & \sigma^2{}_{i,2} & 0 & \cdots & 0 & 0 \\ 0 & 0 & \sigma^2{}_{i,3} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma^2{}_{i,m-1} & 0 \\ 0 & 0 & 0 & \cdots & 0 & \sigma^2{}_{i,m} \end{pmatrix}$$

Iterate. On iteration t let our estimates be

$\lambda_t = \{ \mu_1^{(t)}, \mu_2^{(t)} \ldots \mu_k^{(t)}, \Sigma_1^{(t)}, \Sigma_2^{(t)} \ldots \Sigma_k^{(t)}, p_1^{(t)}, p_2^{(t)} \ldots p_k^{(t)} \}$ $\quad$ $p_i^{(t)} = p^{(t)}(y=i)$

## 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 Max. like **μ** given our data's class membership distributions

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

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

*m* = #data points

# EM for general GMMs

Iterate. On iteration t let our estimates be

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

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

**E-step**

Compute "expected" classes of all datapoints for each class

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

*Just evaluate a Gaussian at $x_j$*

**M-step**

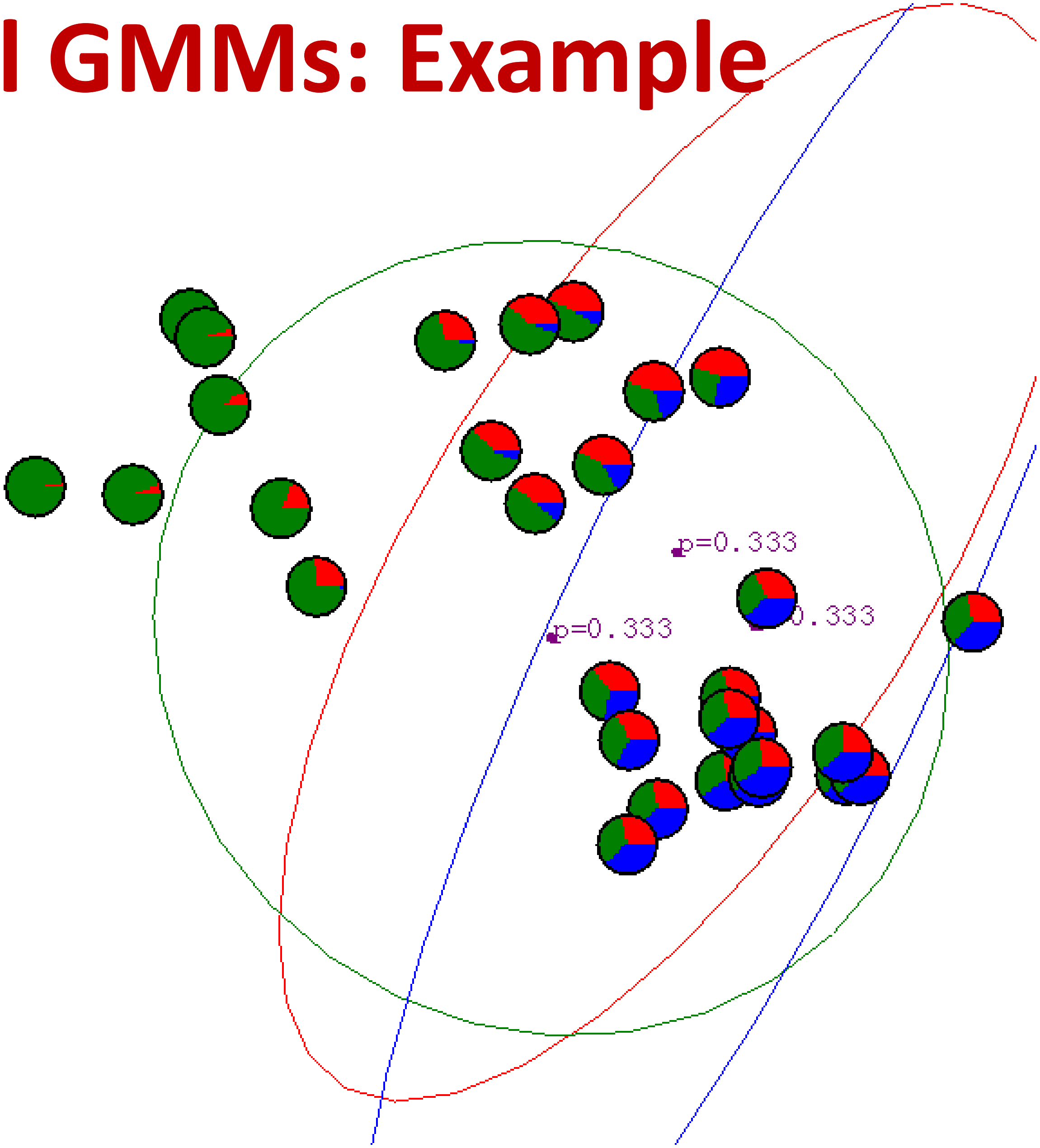
Compute MLEs given our data's class membership distributions (weights)

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

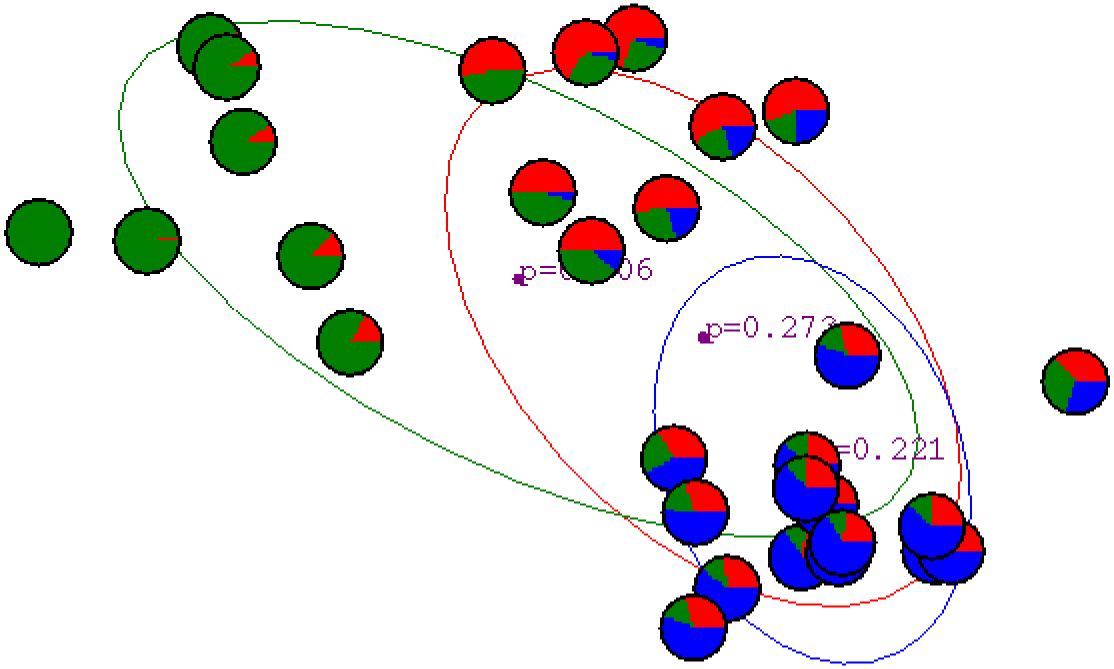
$$p_i^{(t+1)} = \frac{\sum_j P\left(y = i \middle| x_j, \lambda_t\right)}{m}$$

*m* = #data points

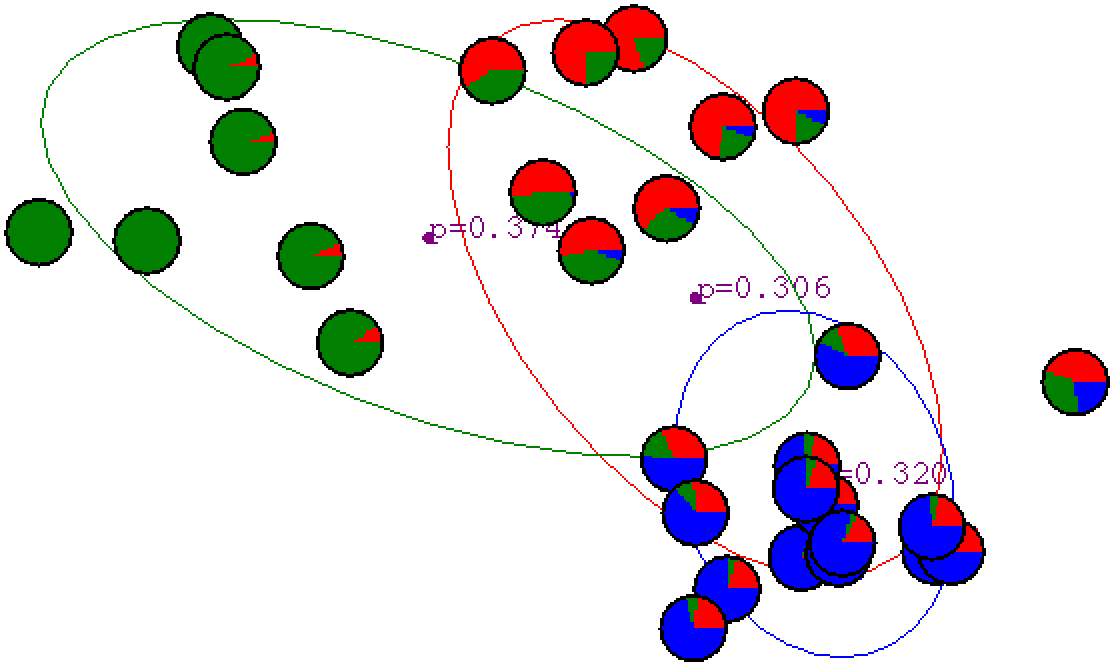
# EM for general GMMs: Example



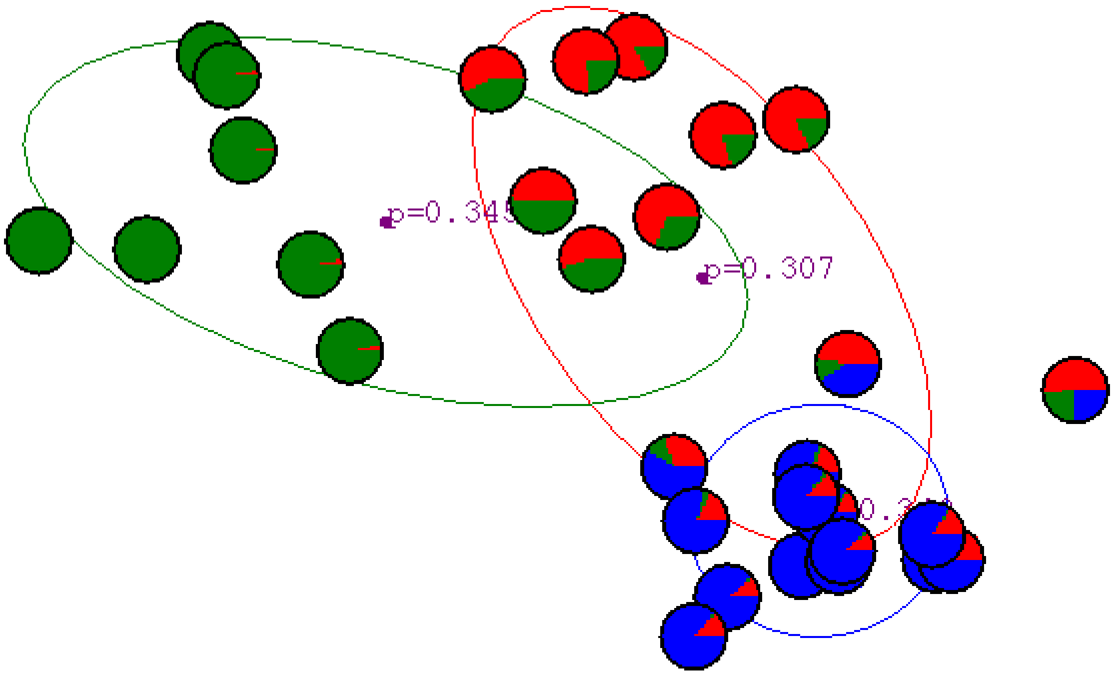
# After 1st iteration



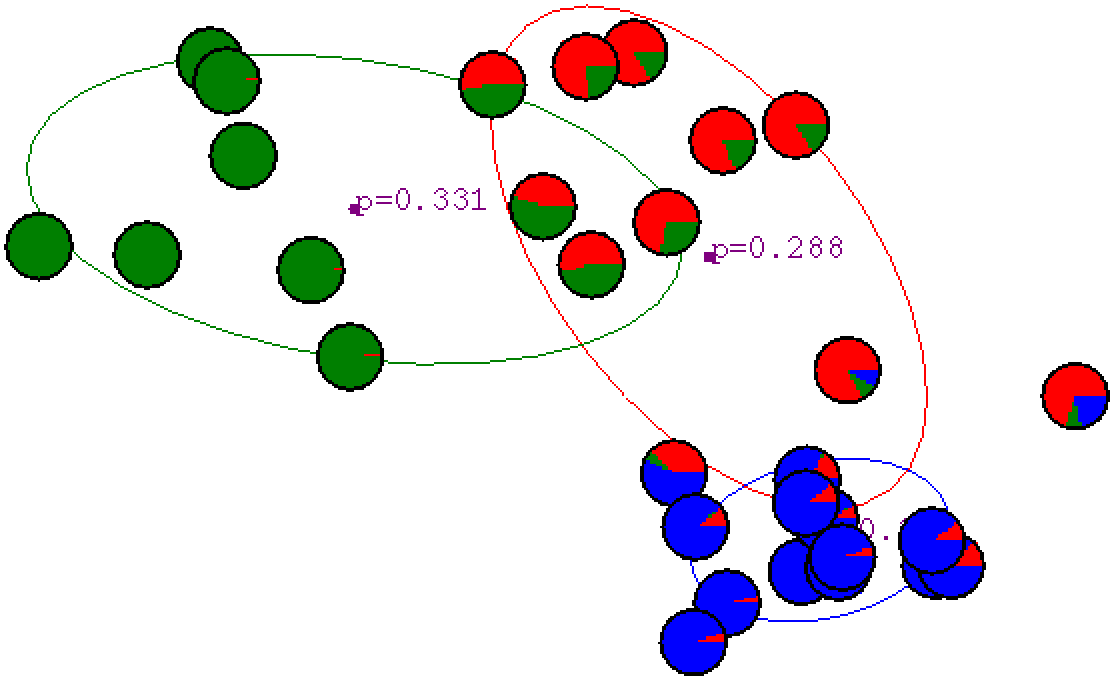
# After 2nd iteration



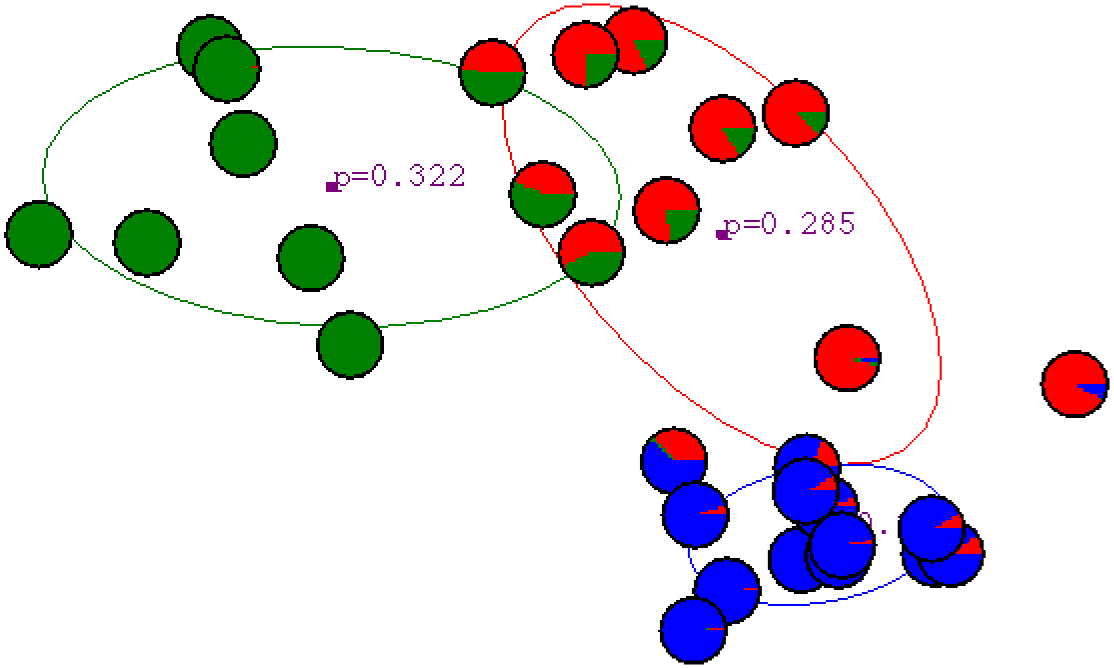
# After 3rd iteration



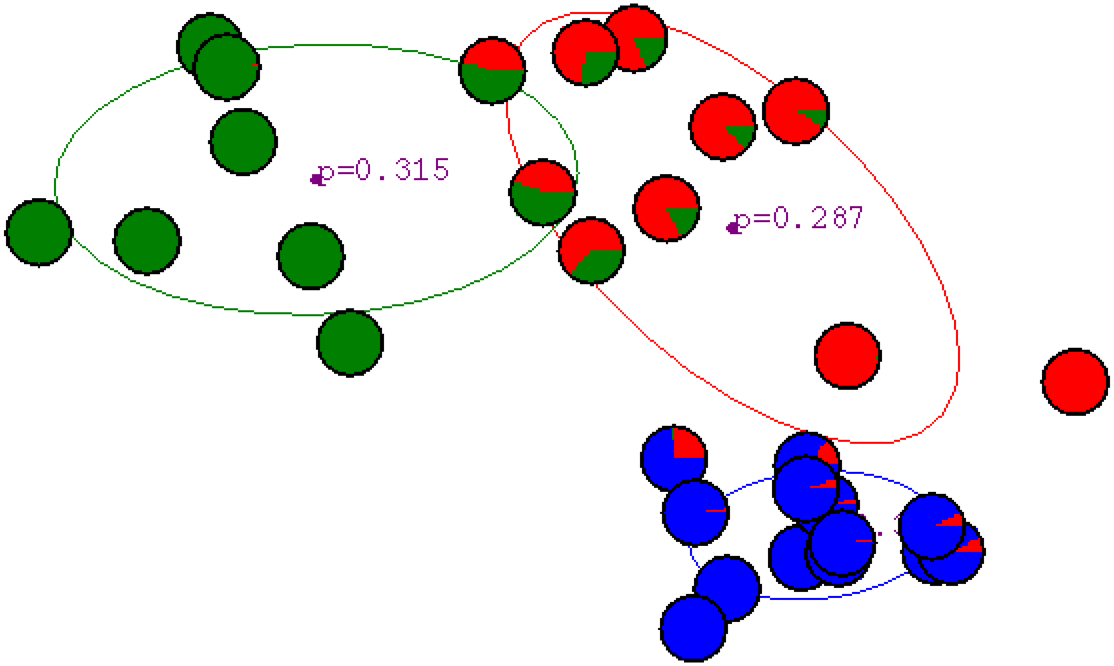
# After 4th iteration



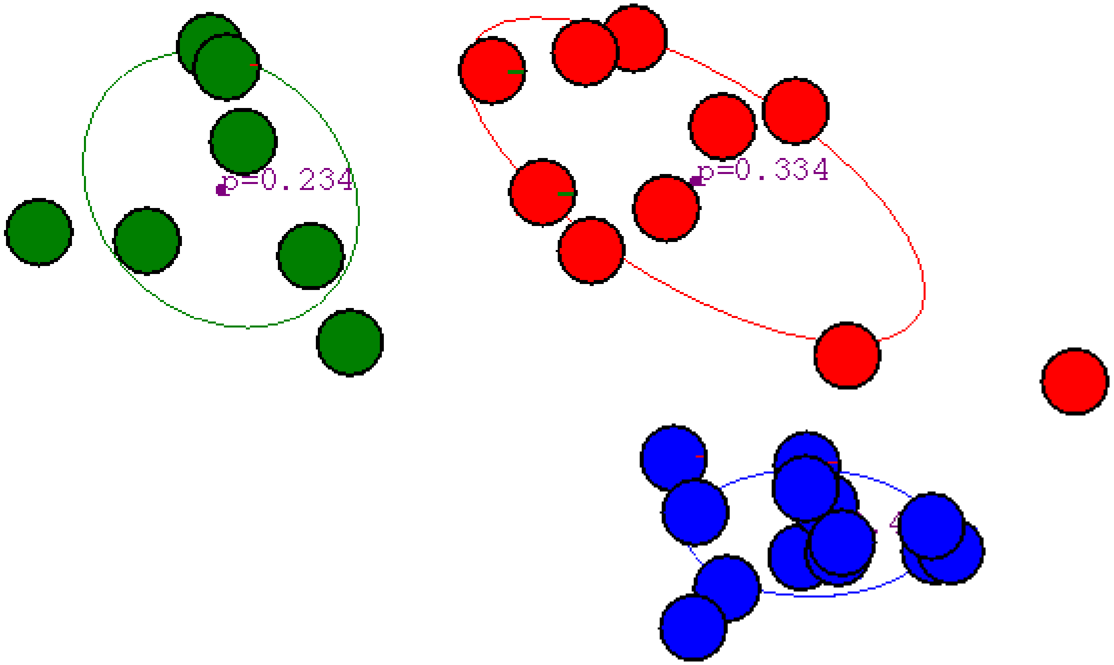
# After 5th iteration



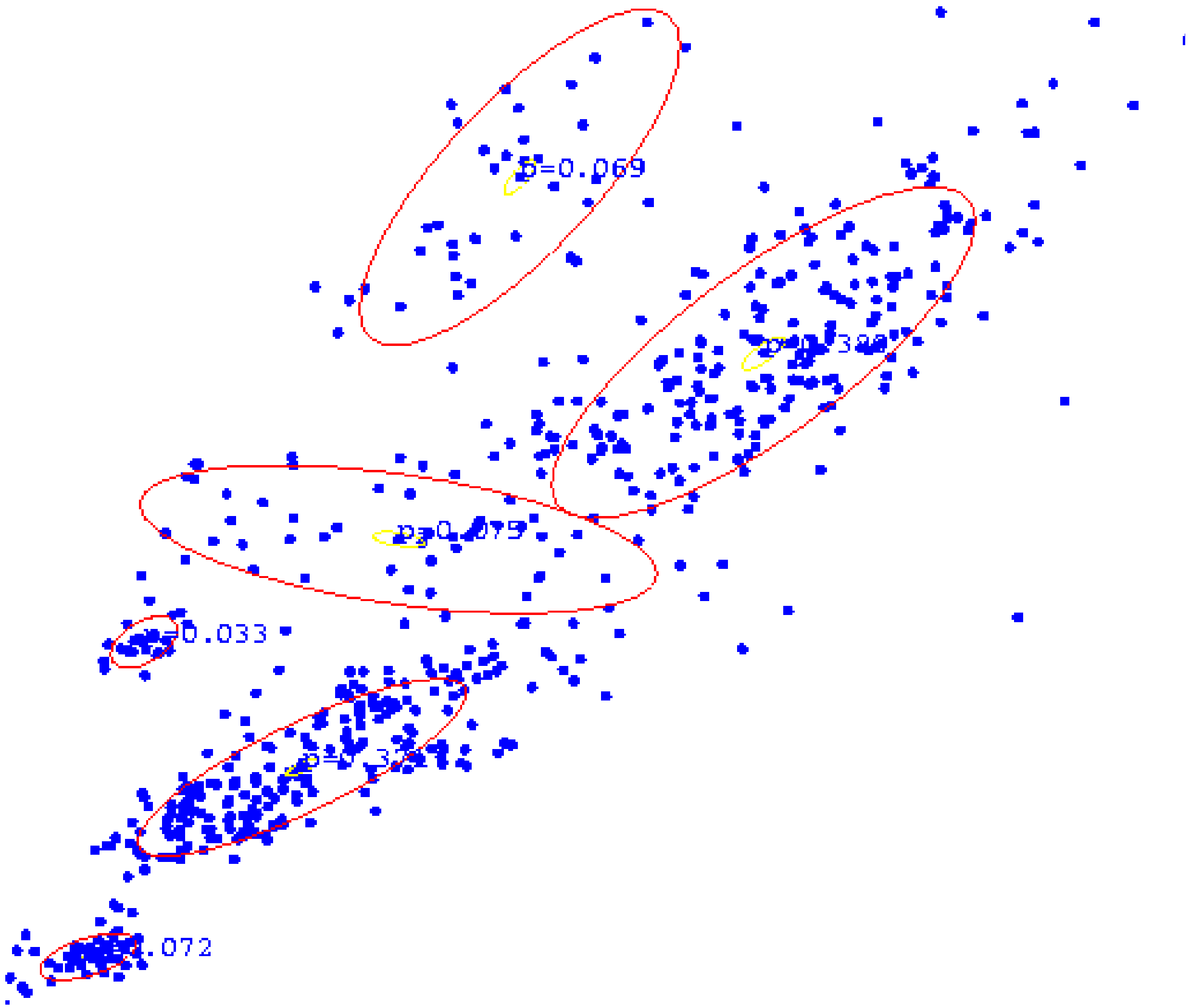
# After 6th iteration



# After 20th iteration

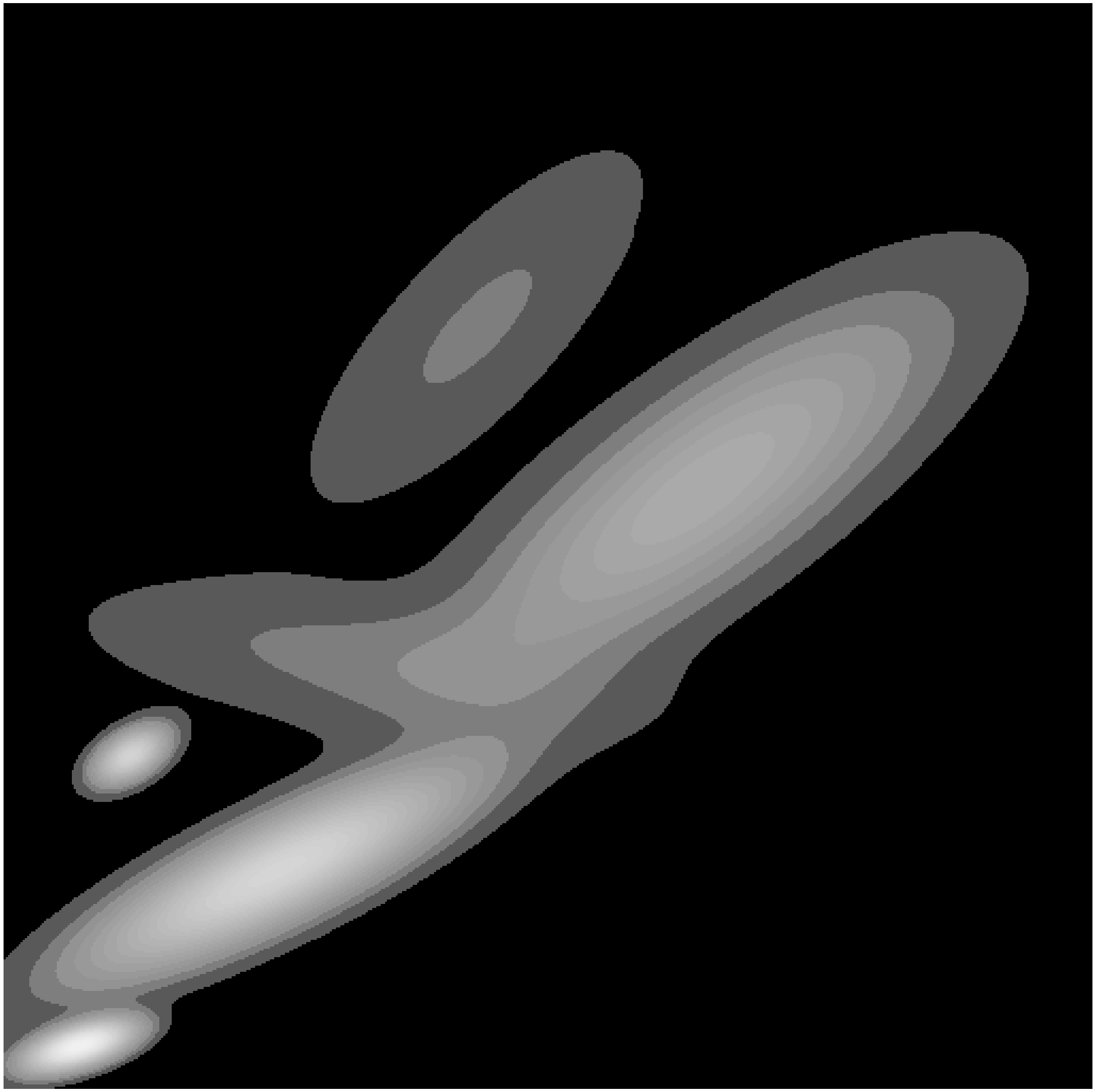


# GMM clustering of the assay data



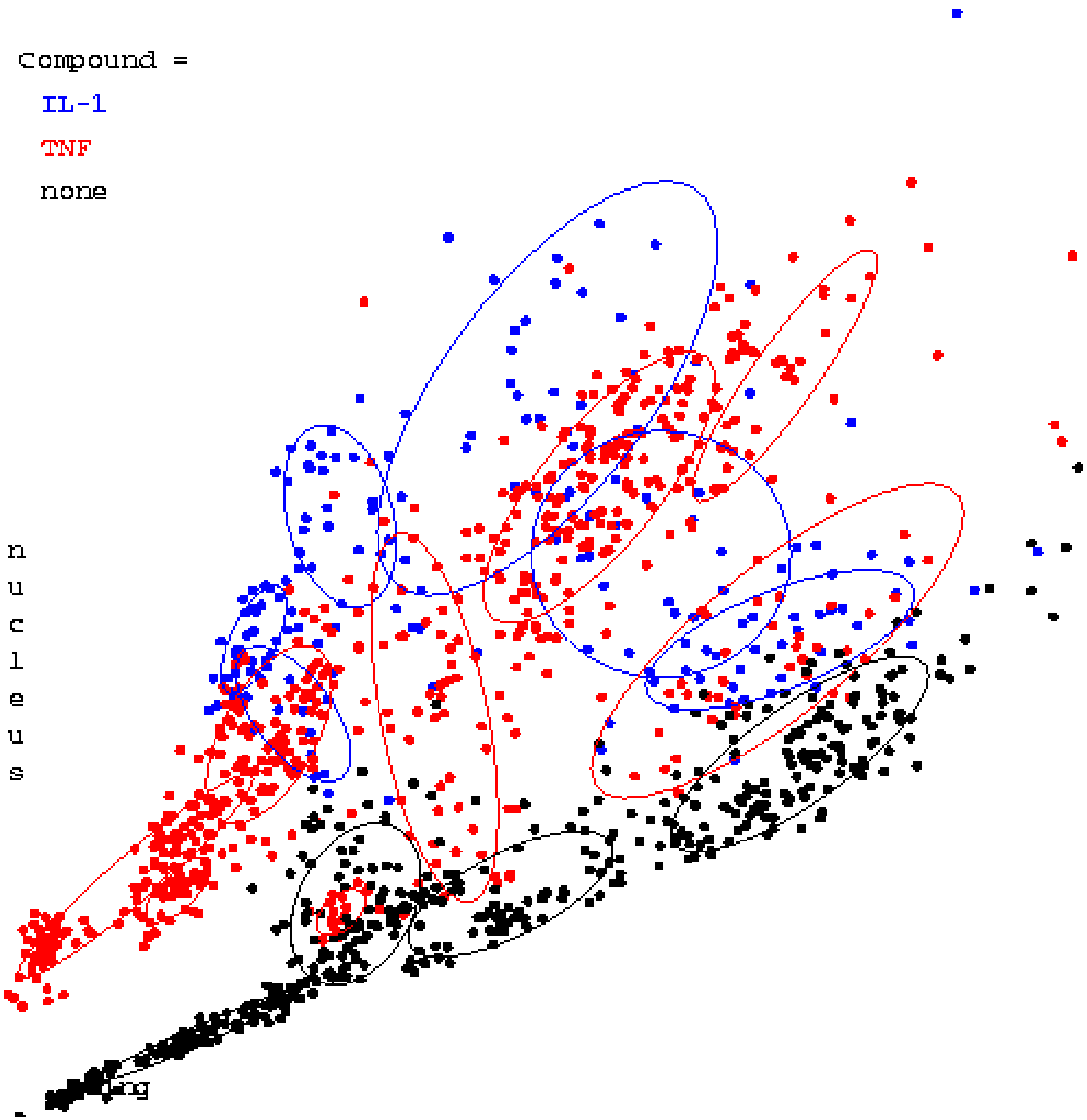
©2005-2009 Carlos Guestrin

# Resulting Density Estimator

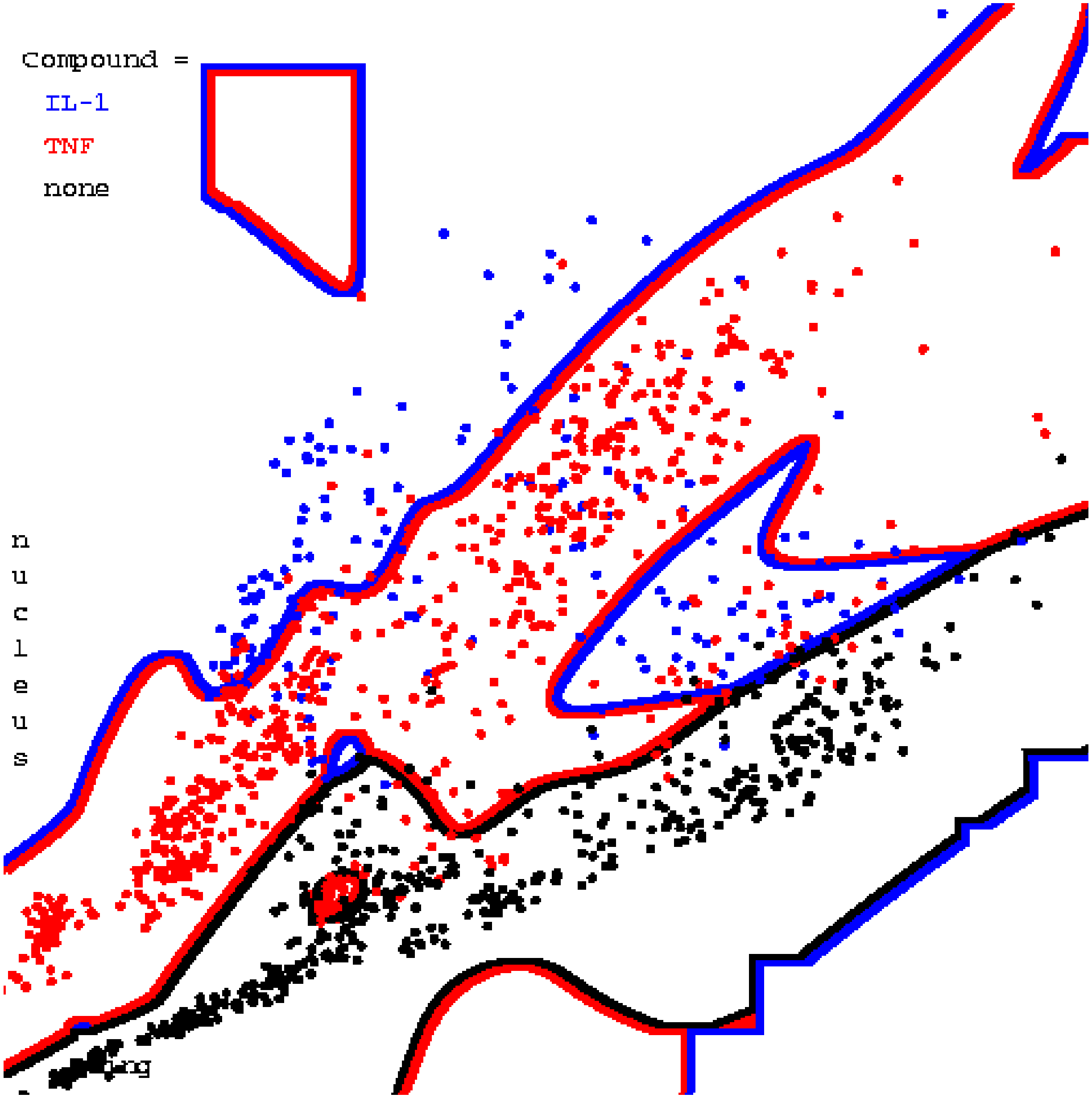


# Three classes of assay

(each learned with it’s own mixture model)



# Resulting Bayes Classifier



# General EM algorithm

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

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

# E step

**x** is observed, **z** is missing

Compute probability of missing data given current choice of $\theta$

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

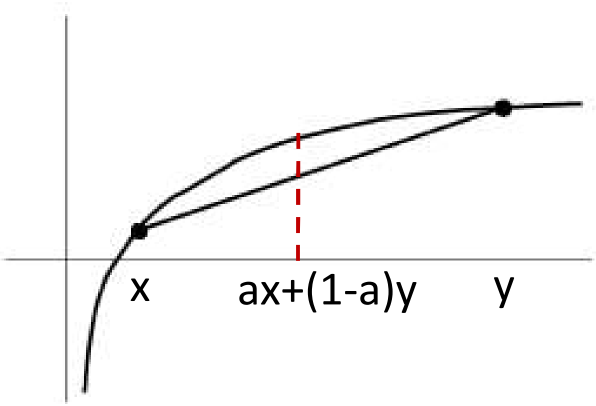
$$\text{E.g., } \mathrm{P}\left(\mathrm{y} = \mathrm{i} \middle| \mathrm{x}_{\mathrm{j}}, \lambda_{\mathrm{t}}\right)$$

# 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:** $\log \sum_{\mathbf{z}} \mathrm{P}(\mathbf{z})\, \mathrm{f}(\mathbf{z}) \geq \sum_{\mathbf{z}} \mathrm{P}(\mathbf{z}) \log \mathrm{f}(\mathbf{z})$



log: concave function

$\log(ax+(1-a)y) \geq a\log(x) + (1-a)\log(y)$

# 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:** $\log \sum_{\mathbf{z}} \mathrm{P}(\mathbf{z})\, \mathrm{f}(\mathbf{z}) \geq \sum_{\mathbf{z}} \mathrm{P}(\mathbf{z}) \log \mathrm{f}(\mathbf{z})$

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

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

# M step

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

Maximize lower bound on marginal likelihood

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

Use expected counts instead of counts:

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

# Convergence of EM

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

**M-step**: Fix Q, maximize F over $\theta$

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

Maximizes lower bound F on marginal likelihood

**E-step**: Fix $\theta$, maximize F over Q

# 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

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

**E-step**: Fix θ, maximize F over Q

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

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

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

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

# Convergence of EM

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

**M-step**: Fix Q, maximize F over θ

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

Maximizes lower bound F on marginal likelihood

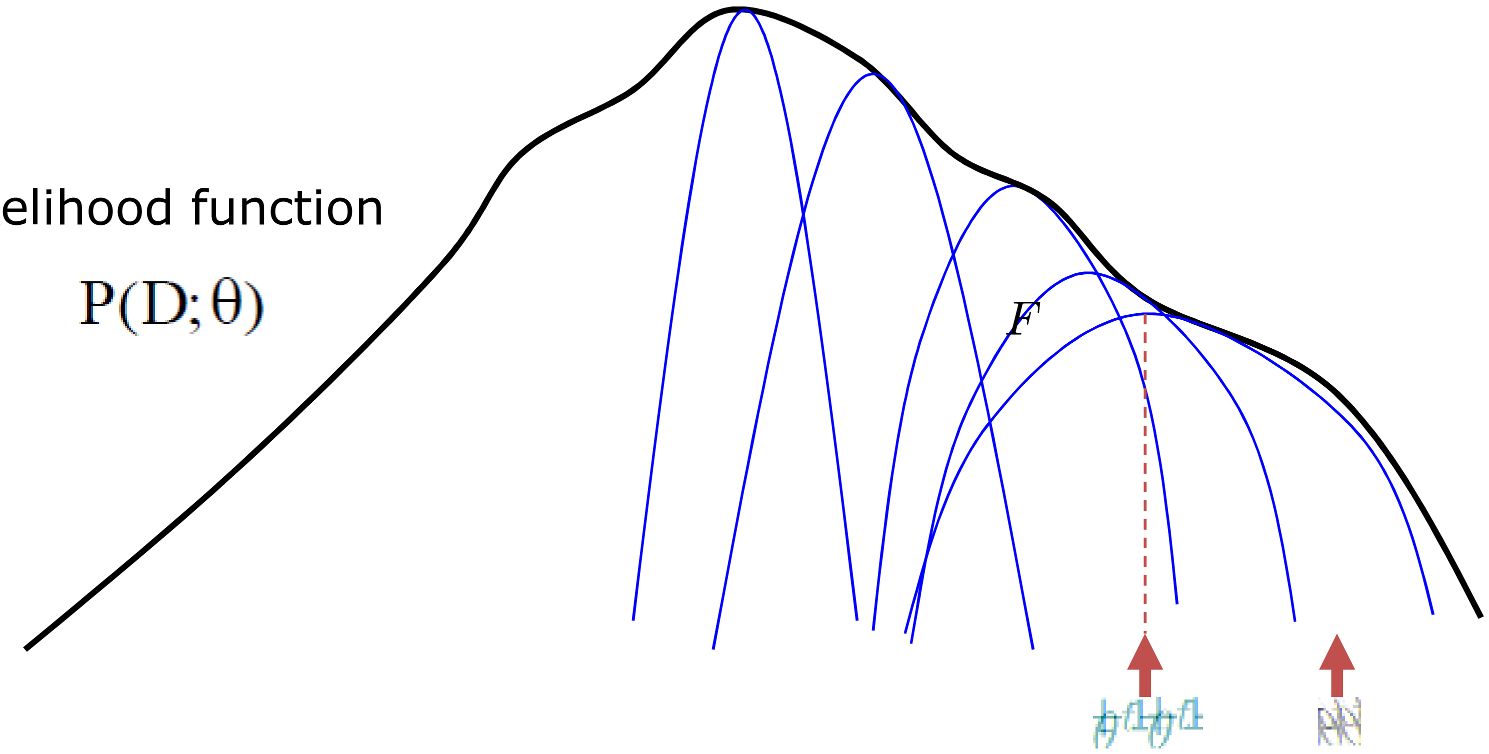
**E-step**: Fix θ, maximize F over Q

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

Re-aligns F with marginal likelihood

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

# Monotonic convergence of EM



Likelihood function

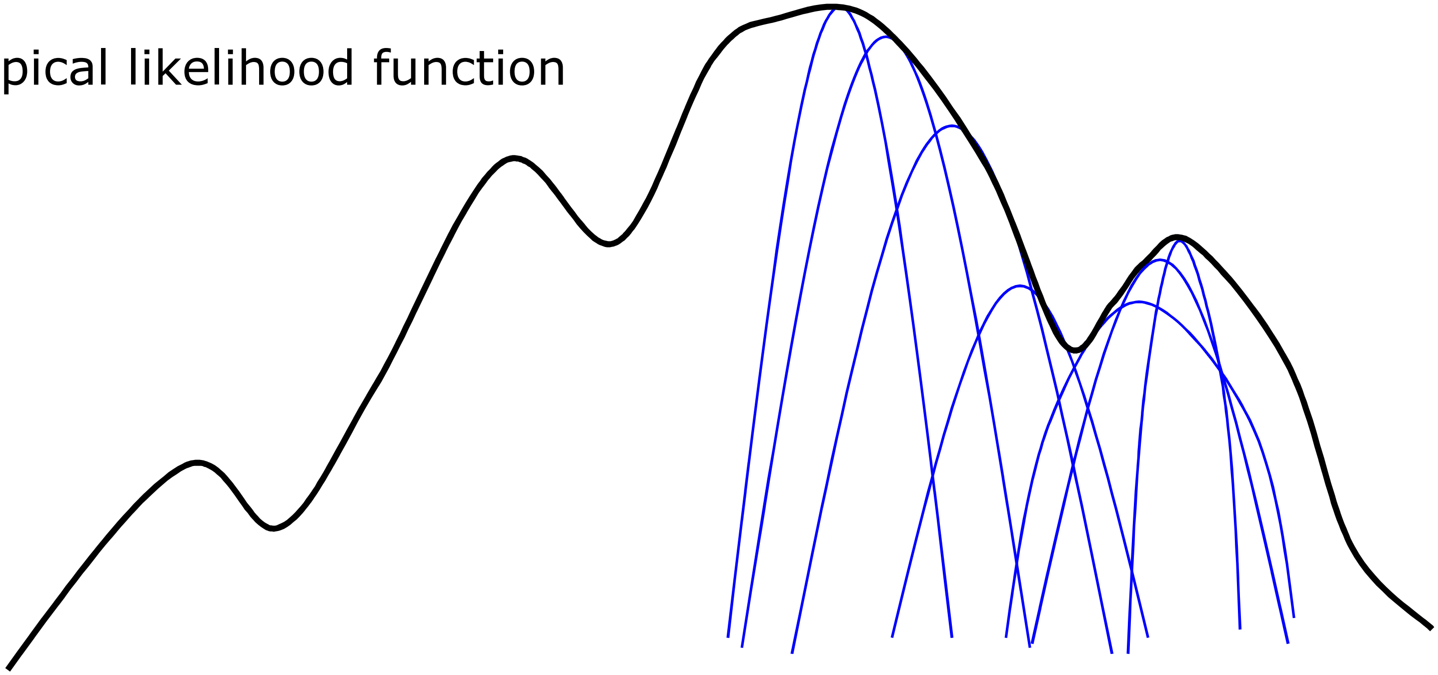
$\mathrm{P}(\mathrm{D};\theta)$

Sequence of EM surrogate $F$-functions

**EM monotonically converges to a local maximum of likelihood !**

# Monotonic convergence of EM

Typical likelihood function



Different sequence of EM surrogate
$F$-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.