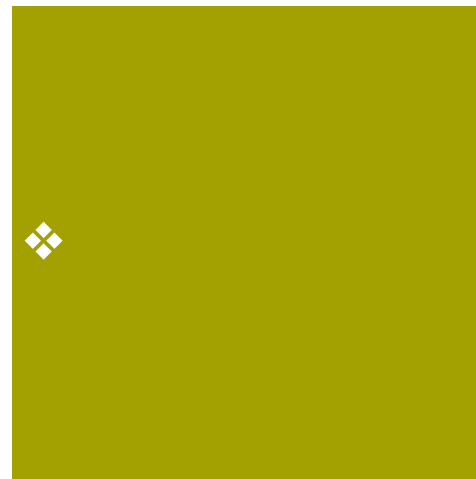




Gaussian Mixture Models & Expectation  
Maximisation Algorithm



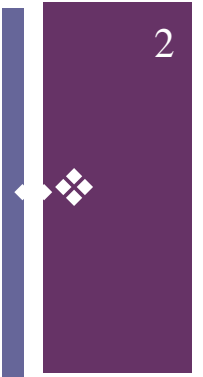
## Pattern Recognition and Image Analysis

Dr. Manal Helal – Fall 2015

Lecture 5

# Objectives

- GMM
- EM



- In previous lectures we showed how to build classifiers when the underlying densities are known
- In most situations, however, the true distributions are unknown and must be estimated from data
- – Two approaches are common

## 1. Parameter Estimation (Covered)

- Assume a particular form for the density (e.g. Gaussian), so only the parameters (e.g., mean and variance) need to be estimated
  - Maximum Likelihood
  - Bayesian Estimation

## 2. Non-parametric Density Estimation (the next two lectures)

- Kernel Density Estimation (Lecture 6)
- Nearest Neighbour Rule

# Gaussian Mixture

The Gaussian mixture architecture estimates probability density functions (PDF) for each class, and then performs classification based on Bayes' rule:

$$\hat{P}(C_i | X) = P(X | C_i) \cdot \frac{P(C_i)}{P(X)}$$

Where  $P(X | C_i)$  is the PDF of class  $j$ , evaluated at  $X$ ,  $P(C_j)$  is the prior probability for class  $j$ , and  $P(X)$  is the overall PDF, evaluated at  $X$ .

# Gaussian Mixture

Unlike the unimodal Gaussian architecture, which assumes  $P(X | C_j)$  to be in the form of a Gaussian, the Gaussian mixture model estimates  $P(X | C_j)$  as a weighted average of multiple Gaussians.

$$\P(X | C_j) = \sum_{k=1}^{N_c} w_k G_k$$

Where  $w_k$  is the weight of the k-th Gaussian  $G_k$  and the weights sum to one. One such PDF model is produced for each class.

# Gaussian Mixture

Each Gaussian component is defined as:

$$G_k = \frac{1}{(2\pi)^{n/2} |V_k|^{1/2}} \cdot e^{[-1/2(X-M_k)^T V_k^{-1} (X-M_k)]}$$

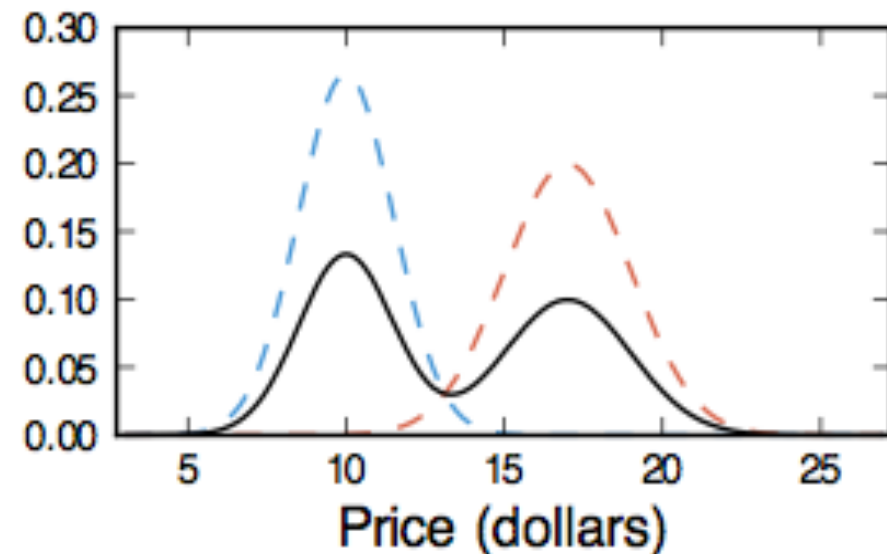
Where  $\mathbf{M}_k$  is the mean of the Gaussian and  $\mathbf{V}_k$  is the covariance matrix of the Gaussian..

# Gaussian Mixture

Free parameters of the Gaussian mixture model consist of the means and covariance matrices of the Gaussian components and the weights indicating the contribution of each Gaussian to the approximation of  $P(X | C_j)$ .

# Example 1

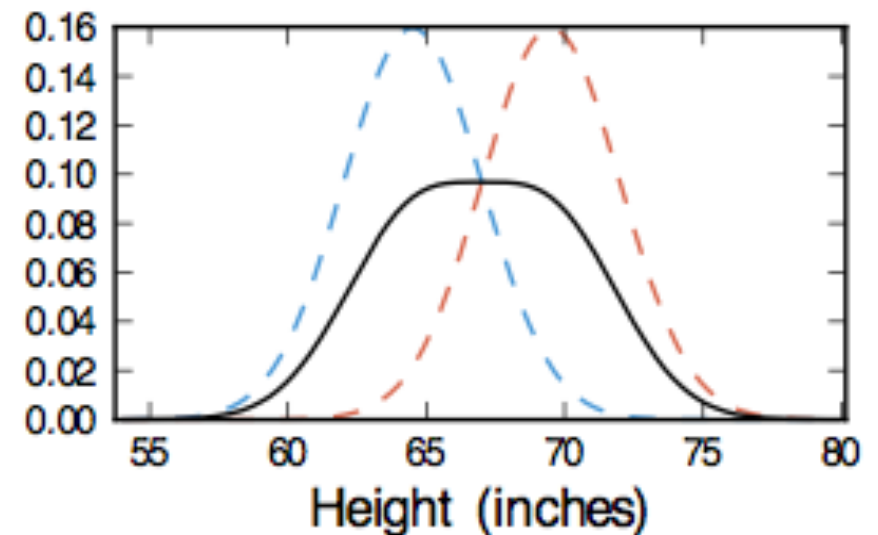
- The price of a randomly chosen paperback book is normally distributed with mean \$10.00 and standard deviation \$1.00
- The price of a randomly chosen hardback is normally distributed with mean \$17 and variance \$1.50
- Is the price of any book selected at random from both groups, will be normally distributed?





## Example 2

- The height of a randomly chosen man is normally distributed with a mean around 5'9.5" and standard deviation around 2.5"
- The height of a randomly chosen woman is normally distributed with a mean around 5'4.5" and standard deviation around 2.5"
- Is the height of any person selected at random from both groups, will be normally distributed?



# Composition of Gaussian Mixture

$$P(C_j | X) = P(X | C_j) \cdot \frac{P(C_j)}{P(X)}$$

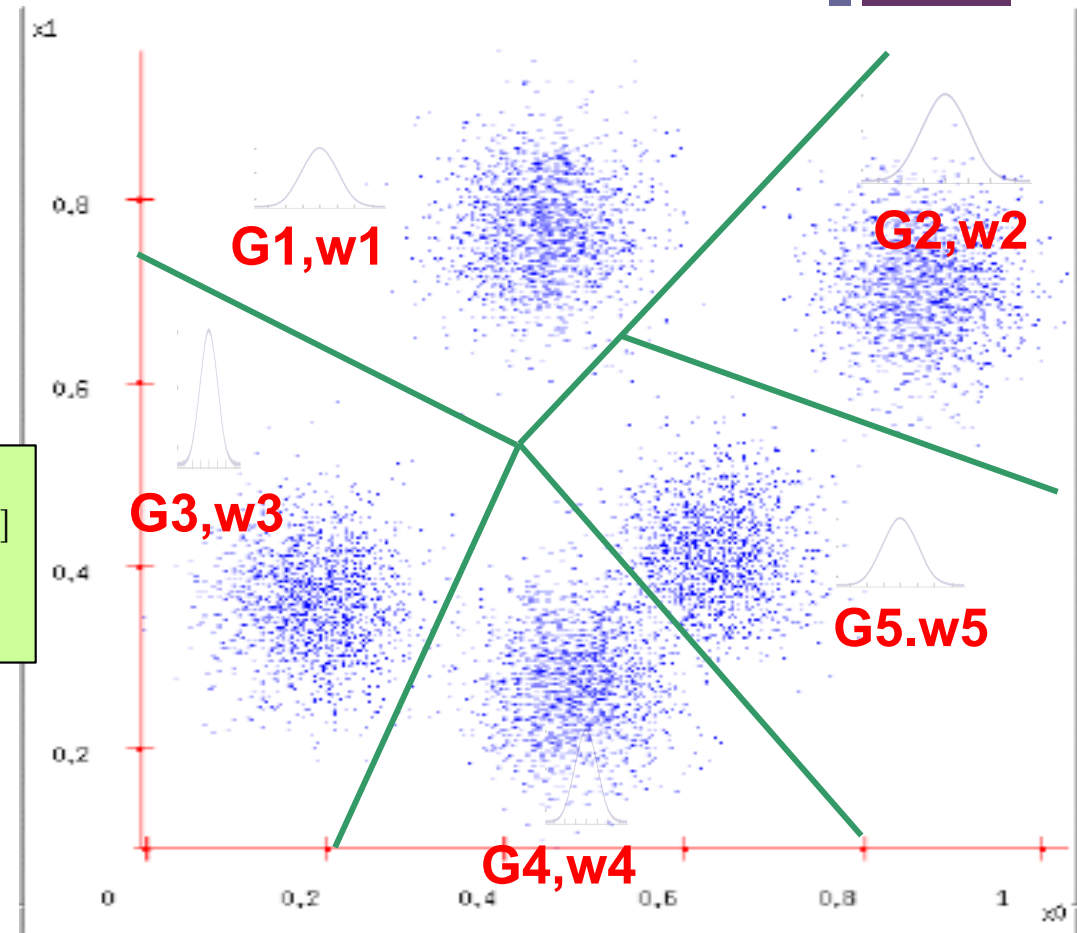
$$P(X | C_j) = \sum_{k=1}^{N_c} w_k G_k$$

$$G_k \equiv p(X | G_i) = \frac{1}{(2\pi)^{d/2} |V_i|^{1/2}} \cdot e^{[-1/2(X-\mu_i)^T V_i^{-1}(X-\mu_i)]}$$

Variables:  $\mu_i$ ,  $V_i$ ,  $w_k$

We use EM (estimate-maximize) algorithm to approximate this variables.

**Class 1**



# Gaussian Mixture

These parameters are tuned using a complex iterative procedure called the estimate-maximise (EM) algorithm, that aims at maximising the likelihood of the training set generated by the estimated PDF.

The likelihood function  $L$  for each class  $j$  can be defined as:

$$\diamond L_j = \prod_{i=0}^{N_{train}} P(X_i | C_j) \longrightarrow \diamond \ln(L_j) = \sum_{i=0}^{N_{train}} \ln(P(X_i | C_j))$$

# Expectation Maximisation Algorithm – Informal Steps

- Initialise the parameters somehow.
- First, fix the parameters (in this case, the means  $\mu_M$  and  $\mu_F$  of the Gaussians) and solve for the posterior distribution for the hidden variables (in this case,  $qC_i$ , the class labels).
- Second, fix the posterior distribution for the hidden variables (again, that's  $qC_i$ , the class labels), and optimise the parameters (the means  $\mu_M$  and  $\mu_F$ ) using the expected values of the hidden variables (in this case, the probabilities from  $qC_i$ ).
- Repeat the two steps above until the values aren't changing much (i.e., until convergence).

# Gaussian Mixture Training Flow Chart (1)

Initialise the initial **Gaussian means**  $\mu_i$ ,  $i=1, \dots, G$  using the K means clustering algorithm

Initialise the **covariance matrices**,  $V_i$ , to the distance to the nearest cluster.

Initialise the **weights**  $\pi_i = 1 / G$  so that all Gaussian are equally likely.



Present each pattern  $X$  of the training set and model each of the classes  $K$  as a weighted sum of Gaussians:

$$p(X | \theta_s) = \sum_{i=1}^G \pi_i p(X | G_i)$$

Where **G** is the number of Gaussians, the  **$\pi_i$** 's are the weights, and

$$p(X | G_i) = \frac{1}{(2\pi)^{d/2} |V_i|^{1/2}} \cdot e^{[-1/2(X - \mu_i)^T V_i^{-1} (X - \mu_i)]}$$

Where  **$V_i$**  is the covariance matrix.



# Gaussian Mixture Training Flow Chart (2)

Compute:

$$\tau_{ip} \equiv P(G_i | X) = \frac{\pi_i p(X | G_i, C_k)}{p(X)} = \frac{\pi_i p(X | G_i, C_k)}{\sum_{j=1}^G \pi_j p(X | \theta_j, C_k)}$$

Iteratively update the **weights**, **means** and **covariances**:

$$\pi_i(t+1) = \frac{1}{N_c} \sum_{p=1}^{N_c} \tau_{ip}(t)$$

$$\mu_i(t+1) = \frac{1}{N_c \pi_i(t)} \sum_{p=1}^{N_c} \tau_{ip}(t) X_p$$

$$V_i(t+1) = \frac{1}{N_c \pi_i(t)} \sum_{p=1}^{N_c} \tau_{ip}(t) ((X_p - \mu_i(t))(X_p - \mu_i(t))^T)$$

## Gaussian Mixture Training Flow Chart (3)



Recompute  $\tau_{ip}$  using the new weights, means and covariances. Stop training if

$$\Delta\tau_{ip} \equiv \tau_{ip}(t+1) - \tau_{ip}(t) \leq \textit{threshold}$$

Or the number of epochs reach the specified value. Otherwise, continue the iterative updates.



# Gaussian Mixture Test Flow Chart

Present each input pattern  $X$  and compute the confidence for each class  $j$ :

$$P(C_j)P(X | \theta_x, C_j)$$

Where  $P(C_j) = \frac{N_{ci}}{N}$  is the prior probability of class  $C_j$  estimated by counting the number of training patterns. Classify pattern  $X$  as the class with the highest confidence.



# Preliminaries: Entropy

“I thought of calling it "information", but the word was overly used, so I decided to call it "uncertainty". [...] Von Neumann told me, "You should call it entropy, for two reasons. In the first place your uncertainty function has been used in statistical mechanics under that name, so it already has a name. In the second place, and more important, nobody knows what entropy really is, so in a debate you will always have the advantage.”

- Conversation between Claude Shannon and John von Neumann regarding what name to give to the attenuation in phone-line signals.

# Shannon Entropy

- **Entropy** is the average amount of information contained in each message received. Here, *message* stands for an event, sample or character drawn from a distribution or data stream.
- Entropy thus characterises our uncertainty about our source of information. The idea here is that the less likely an event is, the more information it provides when it occurs.
- Shannon Entropy  $H(X)$  for random variable  $X$  with Probability distribution  $P(X)$  is defined as:

$$H(X) = \sum_i P(x_i) I(x_i) = - \sum_i P(x_i) \log_b P(x_i)$$

$I$  is the information content of  $X$ .  $I(X)$  is itself a random variable, defined as the negative of the logarithm of the probability distribution.

# Preliminaries: KL Divergence

- The Kullback–Leibler divergence (a.k.a information divergence, information gain, relative entropy, or KLIC) is a non-symmetric measure of the difference between two probability distributions  $P$  and  $Q$ .
- The KL divergence of  $Q$  from  $P$ , denoted  $D_{\text{KL}}(P\|Q)$ , is a measure of the information lost when  $Q$  is used to approximate  $P$ .
- $D_{\text{KL}}(P\|Q)$  is not symmetric to  $D_{\text{KL}}(Q\|P)$

- For Discrete Probability: 
$$D_{\text{KL}}(P\|Q) = \sum_i P(i) \ln \frac{P(i)}{Q(i)}.$$

i.e. it is the expectation of the logarithmic difference between the probabilities  $P$  and  $Q$ , where the expectation is taken using the probabilities  $P$ . The KL divergence is only defined if  $P$  and  $Q$  both sum to 1 and if  $Q(i) > 0$  implies  $P(i) > 0$  for all  $i$

- For Continuous Probability:

where  $p$  and  $q$  denote the densities of  $P$  and  $Q$ .

$$D_{\text{KL}}(P\|Q) = \int_{-\infty}^{\infty} p(x) \ln \frac{p(x)}{q(x)} dx,$$

# Preliminaries: Jensen's Inequality

- Generally it relates the value of a convex function of an integral to the integral of the convex function. i.e. the convex transformation of a mean is less than or equal to the mean after convex transformation; it is a simple corollary that the opposite is true of concave transformations.
- In our case, we need this form:

$$\log(\mathbb{E}[X]) \geq \mathbb{E}[\log(X)]$$

- For a geometric intuition and a proof and more detail, see Wikipedia

# Preliminaries: Marginal Distribution

- The marginal distribution of a subset of a collection of random variables is the probability distribution of the variables contained in the subset. It gives the probabilities of various values of the variables in the subset without reference to the values of the other variables. This contrasts with a conditional distribution, which gives the probabilities contingent upon the values of the other variables.

Joint and marginal distributions of a pair of discrete, random variables  $X, Y$  having nonzero mutual information  $I(X; Y)$ . The values of the joint distribution are in the  $4 \times 4$  square, and the values of the marginal distributions are along the right and bottom margins.

	$x_1$	$x_2$	$x_3$	$x_4$	$p_Y(Y) \downarrow$
$y_1$	$\frac{4}{32}$	$\frac{2}{32}$	$\frac{1}{32}$	$\frac{1}{32}$	$\frac{8}{32}$
$y_2$	$\frac{2}{32}$	$\frac{4}{32}$	$\frac{1}{32}$	$\frac{1}{32}$	$\frac{8}{32}$
$y_3$	$\frac{2}{32}$	$\frac{2}{32}$	$\frac{2}{32}$	$\frac{2}{32}$	$\frac{8}{32}$
$y_4$	$\frac{8}{32}$	0	0	0	$\frac{8}{32}$
$p_X(X) \rightarrow$	$\frac{16}{32}$	$\frac{8}{32}$	$\frac{4}{32}$	$\frac{4}{32}$	$\frac{32}{32}$

# Preliminaries: Marginal Distribution –

## Cont'd

- Given two random variables  $X$  and  $Y$  whose joint distribution is known, the marginal distribution of  $X$  is simply the probability distribution of  $X$  averaging over information about  $Y$ . It is the probability distribution of  $X$  when the value of  $Y$  is not known. This is typically calculated by summing or integrating the joint probability distribution over  $Y$ .
- For Discrete Random Variables:

$$\Pr(X = x) = \sum_y \Pr(X = x, Y = y) = \sum_y \Pr(X = x|Y = y) \Pr(Y = y),$$

where  $\Pr(X = x, Y = y)$  is the joint distribution of  $X$  and  $Y$ , while  $\Pr(X = x|Y = y)$  is the conditional distribution of  $X$  given  $Y$

- For Continuous Random Variables:

$$p_X(x) = \int_y p_{X,Y}(x, y) \, dy = \int_y p_{X|Y}(x|y) p_Y(y) \, dy,$$

where  $p_{X,Y}(x, y)$  gives the joint distribution of  $X$  and  $Y$ , while  $p_{X|Y}(x|y)$  gives the conditional distribution for  $X$  given  $Y$ .

# EM Formal Algorithm

- The EM algorithm is actually maximising a lower bound on the log likelihood (in other words, each step is guaranteed to improve our answer until convergence).

# EM Steps

## 1. Maximize the log-likelihood

a. Marginalizing over  $C$  and introducing  $q_C(c)/q_C(c)$

$$\begin{aligned}\log p_Y(y; \theta) &= \log \left( \sum_c q_C(c) \frac{p_{Y,C}(y, c; \theta)}{q_C(c)} \right) \\ &= \log \left( \mathbb{E}_{q_C} \left[ \frac{p_{Y,C}(y, C; \theta)}{q_C(C)} \right] \right) \\ &\geq \mathbb{E}_{q_C} \left[ \log \frac{p_{Y,C}(y, C; \theta)}{q_C(C)} \right]\end{aligned}$$

b. Rewriting as an expectation

c. Using Jensen's inequality

## 2. M-Step:

a. Rearrange

$$\mathbb{E}_{q_C} \left[ \log \frac{p_{Y,C}(y, C; \theta)}{q_C(C)} \right] = \mathbb{E}_{q_C} [\log p_{Y,C}(y, C; \theta)] - \mathbb{E}_{q_C} [\log q_C(C)]$$

b. Maximizing with respect to  $\theta$ :

$$\hat{\theta} \leftarrow \operatorname{argmax}_{\theta} \mathbb{E}_{q_C} [\log p_{Y,C}(y, C; \theta)]$$

## 3. E-Step:

a. Rearrange

$$\begin{aligned}\mathbb{E}_{q_C} \left[ \log \frac{p_{Y,C}(y, C; \theta)}{q_C(C)} \right] &= \mathbb{E}_{q_C} \left[ \log \frac{p_Y(y; \theta) p_{C|Y}(C|y; \theta)}{q_C(C)} \right] \\ &= \log p_Y(y; \theta) - \mathbb{E}_{q_C} \left[ \log \frac{q_C(C)}{p_{C|Y}(C|y; \theta)} \right]\end{aligned}$$

b. Maximizing with respect to  $q_C$ :

$$\begin{aligned}&= \log p_Y(y; \theta) - D(q_C(\cdot) || p_{C|Y}(\cdot|y; \theta)) \\ \hat{q}_C(\cdot) &\leftarrow p_{C|Y}(\cdot|y; \theta)\end{aligned}$$



# EM Step 1

1. Maximise the log-likelihood:

$$\log p_Y(y; \theta) = \log \left( \sum_c p_{Y,C}(y, c) \right)$$

- a. Log of sum problem!
- b. Solution: if we have an expectation for one variable (C here), we can swap the order using Jensen's inequality.

- c. Introduce a new distribution  $q_C$ :

$$\begin{aligned} & \blacklozenge \log \left( \sum_c q_C(c) \frac{p_{Y,C}(y, c; \theta)}{q_C(c)} \right) \\ &= \log \left( \mathbb{E}_{q_C} \left[ \frac{p_{Y,C}(y, C; \theta)}{q_C(C)} \right] \right) \\ &\geq \mathbb{E}_{q_C} \left[ \log \frac{p_{Y,C}(y, C; \theta)}{q_C(C)} \right] \\ &= \mathbb{E}_{q_C} \left[ \log \frac{p_Y(y; \theta) p_{C|Y}(C|y; \theta)}{q_C(C)} \right] \end{aligned}$$

- a. Using Jensen's inequality :
  - b. Using definition of conditional probability
- Now we have a lower bound on  $\log p_Y(y; \theta)$  that we can optimise pretty easily. Since we've introduced  $q_C$ , we now want to maximise this quantity with respect to both  $\theta$  and  $q_C$ .

# EM Step 2 – The M Step.

2. The M stands for maximisation, since we're maximising with respect to the parameters.
  - a. Find Best Parameters  $\theta$  by rearranging using Jensen's inequality :

$$\mathbb{E}_{q_C} \left[ \log \frac{p_{Y,C}(y, C; \theta)}{q_C(C)} \right] = \mathbb{E}_{q_C} [\log p_{Y,C}(y, C; \theta)] - \mathbb{E}_{q_C} [\log q_C(C)]$$

- b. In general,  $q_C$  doesn't depend on  $\theta$ , so we'll only care about the first term:

$$\hat{\theta} \leftarrow \operatorname{argmax}_{\theta} \mathbb{E}_{q_C} [\log p_{Y,C}(y, C; \theta)]$$

- Now we have a lower bound on  $\log p_Y(y; \theta)$  that we can optimise pretty easily. Since we've introduced  $q_C$ , we now want to maximise this quantity with respect to both  $\theta$  and  $q_C$ .

# EM Step 3 – The E Step.

3. the E stands for expectation, since we're computing  $q_C$  so that we can use it for expectations.
  - a. Find Best  $q_C$  by rearranging using definition of conditional probability:

$$\mathbb{E}_{q_C} \left[ \log \frac{p_Y(y; \theta) p_{C|Y}(C|y; \theta)}{q_C(C)} \right] = \mathbb{E}_{q_C} [\log p_Y(y; \theta)] + \mathbb{E}_{q_C} \left[ \log \frac{p_{C|Y}(C|y; \theta)}{q_C(C)} \right]$$

- a. The first term doesn't depend on  $c$ , and the second term almost looks like a KL divergence:

$$\begin{aligned} &= \log p_Y(y; \theta) - \mathbb{E}_{q_C} \left[ \log \frac{q_C(C)}{p_{C|Y}(C|y; \theta)} \right] \\ &= \log p_Y(y; \theta) - D(q_C(\cdot) || p_{C|Y}(\cdot|y; \theta)) \end{aligned}$$

- b. When maximising this quantity, we want to make the KL divergence as small as possible. KL divergences are always greater than or equal to 0, and they're exactly 0 when the two distributions are equal. So, the optimal  $q_C$  is

$$\begin{aligned} &p_{C|Y}(c|y; \theta) \\ \hat{q}_C(\cdot) &\leftarrow p_{C|Y}(\cdot|y; \theta) \end{aligned}$$

# Repeat Until Convergence

By alternating between

$$\hat{\theta} \leftarrow \operatorname{argmax}_{\theta} \mathbb{E}_{q_C} [\log p_{Y,C}(y, C; \theta)]$$

and

$$\hat{q}_C(\cdot) \leftarrow p_{C|Y}(\cdot|y; \theta)$$

we can maximise a lower bound on the log-likelihood. We've also seen from E-Step that the lower bound is tight (that is, it's equal to the log-likelihood) when we are computing  $q_C$ .

# The EM Algorithm Skeleton

Inputs: Observation  $y$ , joint distribution  $p_{Y,C}(y, c; \psi)$ , conditional distribution  $p_{C|Y}(dy; \psi)$ , initial values  $\psi^{(0)}$

```

1: function EM( $p_{Y,C}(y, c; \psi)$ ,  $p_{C|Y}(dy; \psi)$ ,  $\psi^{(0)}$ )
2:   for iteration  $t = 1, 2, \dots$  do
3:      $q_C^{(t)} \leftarrow p_{C|Y}(dy; \psi^{(t-1)})$  (E-step)
4:      $\psi^{(t)} \leftarrow \operatorname{argmax}_{\psi} E_{q_C^{(t)}} [p_{Y,C}(y, C; \psi)]$  (M-step)
5:     if  $\psi^{(t)} \uparrow \psi^{(t-1)}$  then
6:       return  $\psi^{(t)}$ 

```

# Applying the algorithm for GMM (again)

30

- Given an observed random variable  $Y$  (heights), some hidden variable  $C$  (gender) that  $Y$  depends on. The distributions of  $C$  and  $Y$  have some parameters  $\theta$  (the means  $\mu_M$  and  $\mu_F$ ) that we don't know.
- The objective is to estimate the parameters  $\theta$ , given some initial value: suppose we set  $\mu_M = 3'$  and  $\mu_F = 5'$ . Then the computed posteriors  $q_{C_i}$  would all favour  $F$  over  $M$  (since most people are closer to  $5'$  than  $3'$ ), and we would end up computing  $\mu_F$  as roughly the average of all our heights, and  $\mu_M$  as the average of a few short people.
- For the E-step, we have to compute the posterior distribution  $p_{C_i|Y}$  ( $c_i|y$ ):

- For  $C_i = M$ :

$$\begin{aligned}
 p_{C_i|Y_i}(c_i|y_i) &= \frac{p_{Y_i|C_i}(y_i|c_i)p_{C_i}(c_i)}{p_{Y_i}(y_i)} \\
 &= \frac{\prod_{c \in \{M, F\}} (\uparrow_c N(y_i; \mu_c, \sigma^2))^{\mathbb{I}(c=c_i)}}{\uparrow_M N(y_i; \mu_M, \sigma^2) + \uparrow_F N(y_i; \mu_F, \sigma^2)} = q_{C_i}(c_i) \\
 p_{C_i|Y_i}(M|y_i) &= \frac{\uparrow_M N(y_i; \mu_M, \sigma^2)}{\uparrow_M N(y_i; \mu_M, \sigma^2) + \uparrow_F N(y_i; \mu_F, \sigma^2)} = q_{C_i}(M)
 \end{aligned}$$

# Applying the algorithm for GMM (again)

## – Cont'd

- To do the M-Step:

$$\begin{aligned}
 E_{q_C} [\ln p_{Y,C}(y, C)] &= E_{q_C} [\ln p_{Y|C}(y|C) p_C(C)] \\
 &= E_{q_C} \left[ \sum_{i=1}^n \sum_{c \in \{M, F\}} \left( \frac{1}{c} N(y_i; \mu_c, \sigma^2) \right)^{\mathbb{I}(C_i = c)} \right] \\
 &= E_{q_C} \left[ \sum_{i=1}^n \sum_{c \in \{M, F\}} \mathbb{I}(C_i = c) \left( \ln \frac{1}{c} + \ln N(y_i; \mu_c, \sigma^2) \right) \right] \\
 &= \sum_{i=1}^n \sum_{c \in \{M, F\}} E_{q_C} [\mathbb{I}(C_i = c)] \left( \ln \frac{1}{c} + \ln \frac{1}{\sigma \sqrt{2\pi}} - \frac{(y_i - \mu_c)^2}{2\sigma^2} \right)
 \end{aligned}$$

- $E_{q_C} [\mathbb{I}(C_i = c)]$  is the probability that  $C_i$  is  $c$ , according to  $q$ . Now, we can differentiate with respect to  $\mu_M$  :

$$\frac{d}{d\mu_M} E_{q_C} [\ln p_{Y|C}(y|C) p_C(C)] = \sum_{i=1}^n q_{C_i}(M) \frac{y_i - \mu_M}{\sigma^2} = 0$$

# Applying the algorithm for GMM (again)

## – Cont'd

- Using the last form:

$$\sum_{i=1}^n q_{C_i}(M) \frac{y_i - \mu_M}{\sigma^2} = 0$$

- The solution will be the weighted average as follows:

$$\mu_M = \frac{\sum_{i=1}^n q_{C_i}(M) y_i}{\sum_{i=1}^n q_{C_i}(M)} \quad \mu_F = \frac{\sum_{i=1}^n q_{C_i}(F) y_i}{\sum_{i=1}^n q_{C_i}(F)}$$



# EM Applications

- Filling in **missing data** in samples: When the data indeed has incomplete, missing or corrupted values as a result of a faulty observation process
- Estimating parameters of **finite mixtures**: When assuming the existence of missing or hidden parameters can simplify the likelihood function, which would otherwise lead to an analytically intractable optimisation problem; this is the case we discussed in this lecture
- Estimating the parameters of **HMMs**
- Discovering the value of **latent variables**
- Unsupervised learning of **clusters**, special case for K-Means

# What EM won't do

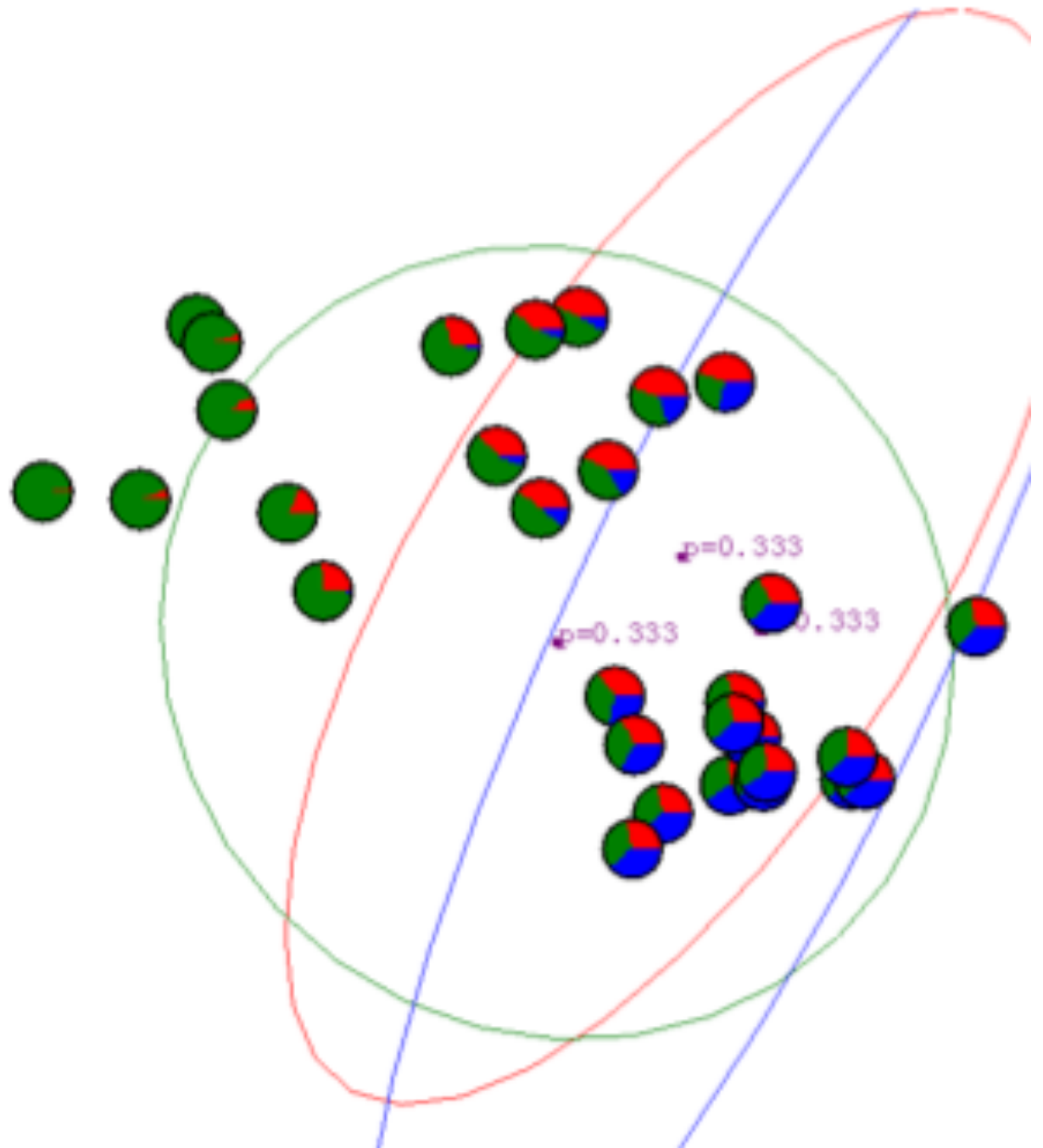
Pick structure of model  
    # components  
    graph structure

Find global maximum

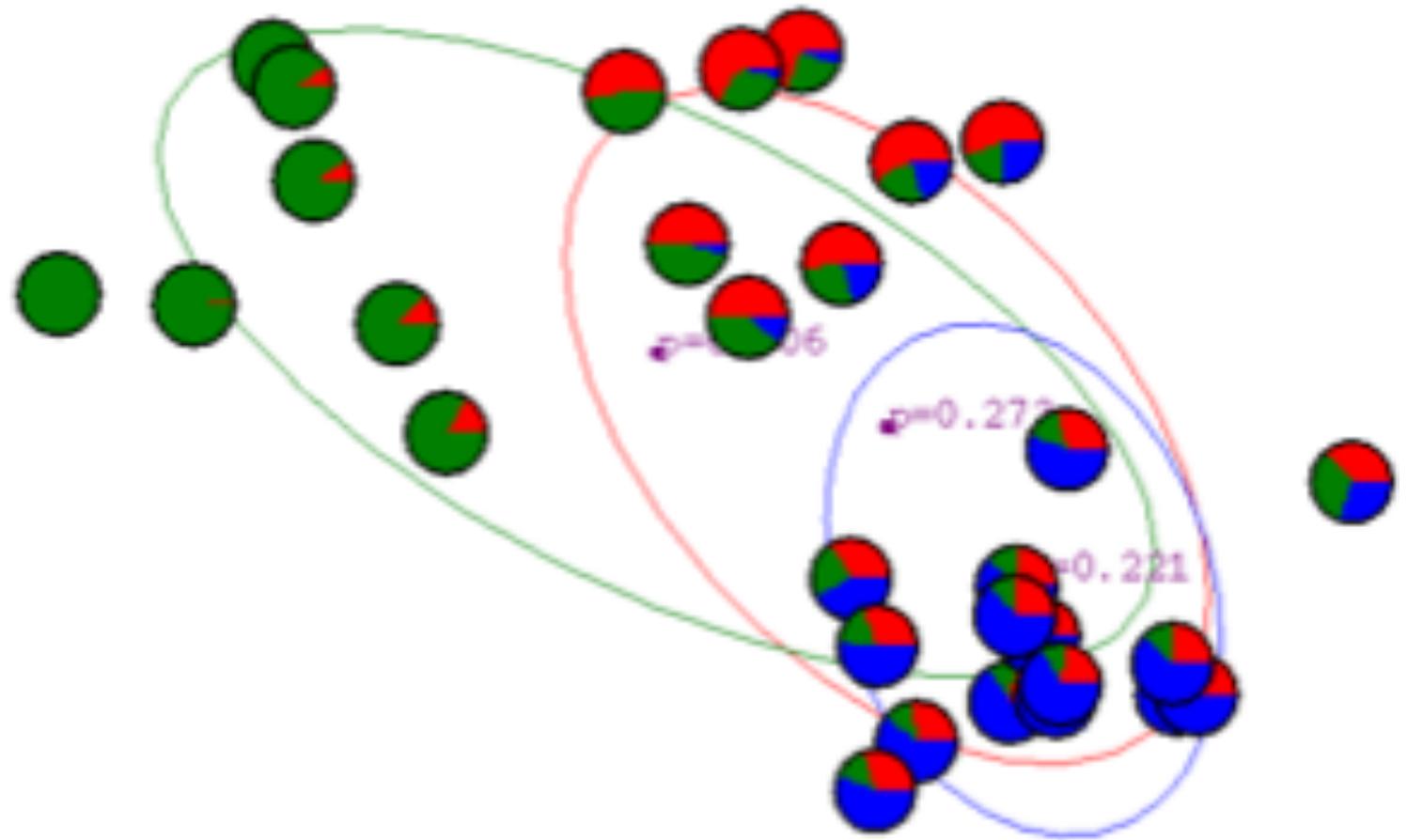
Always have nice closed-form updates  
    optimize within E/M step

Avoid computational problems  
    sampling methods for computing expectations

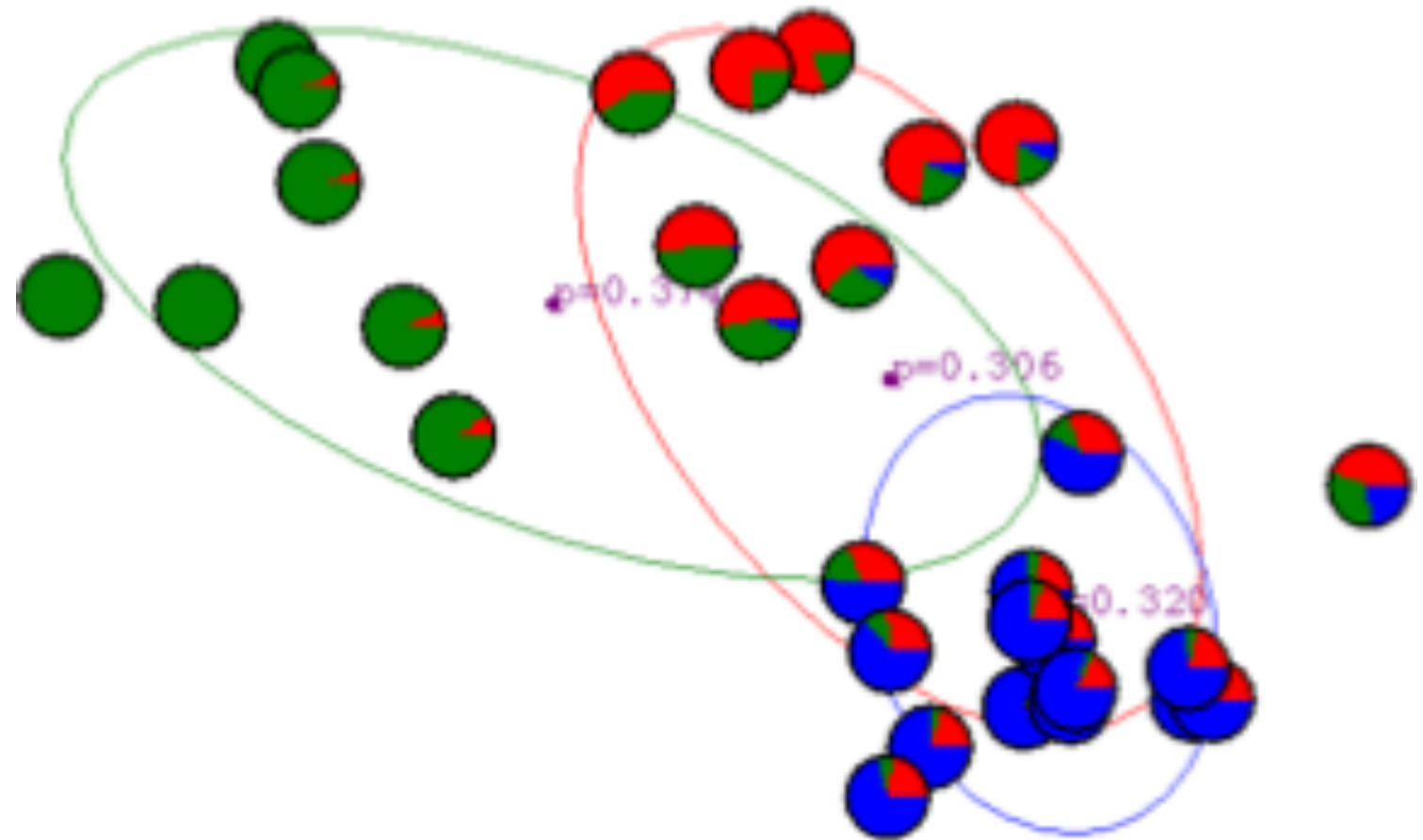
Example:  
Gaussian  
Mixture  
Example:  
Start



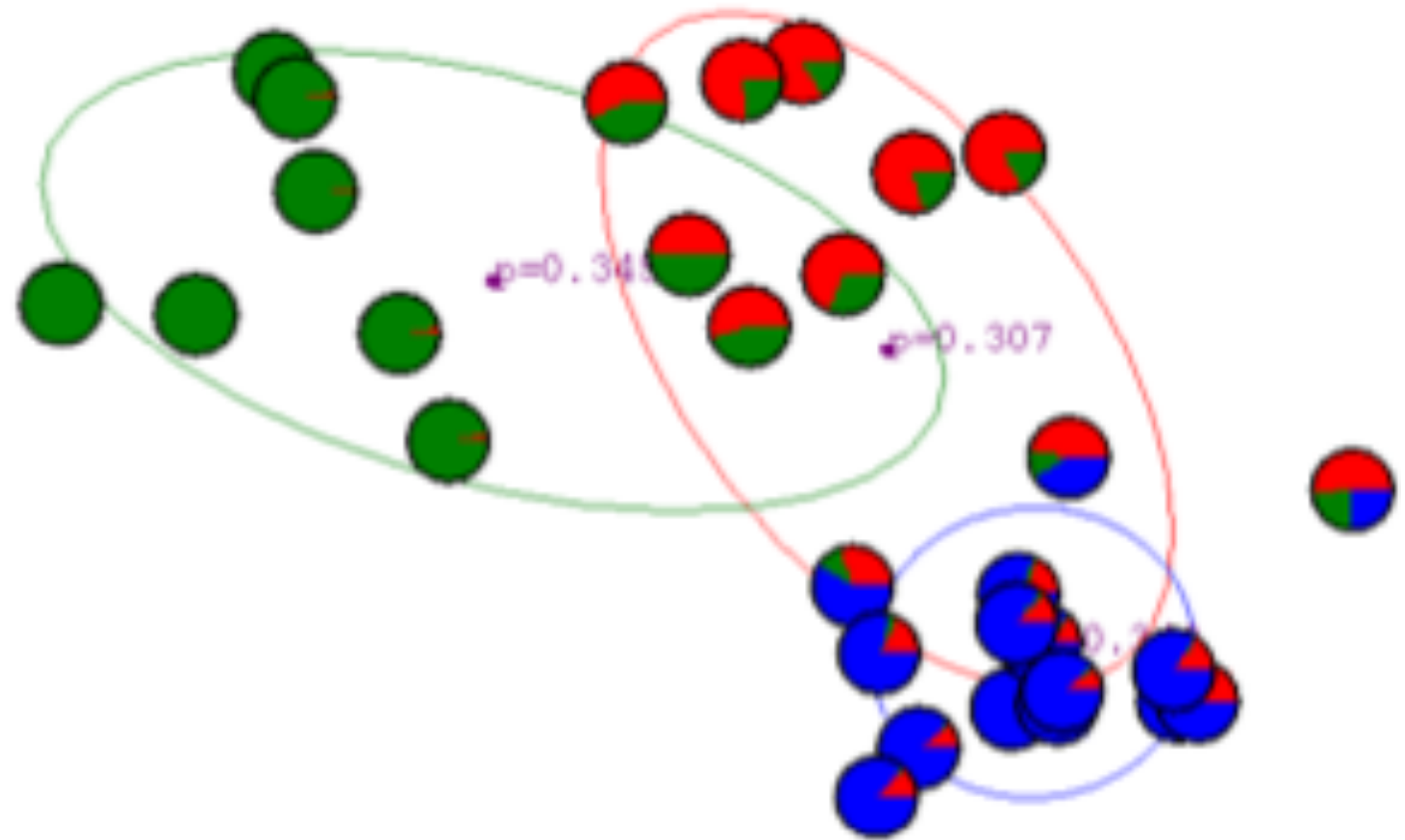
After first  
iteration



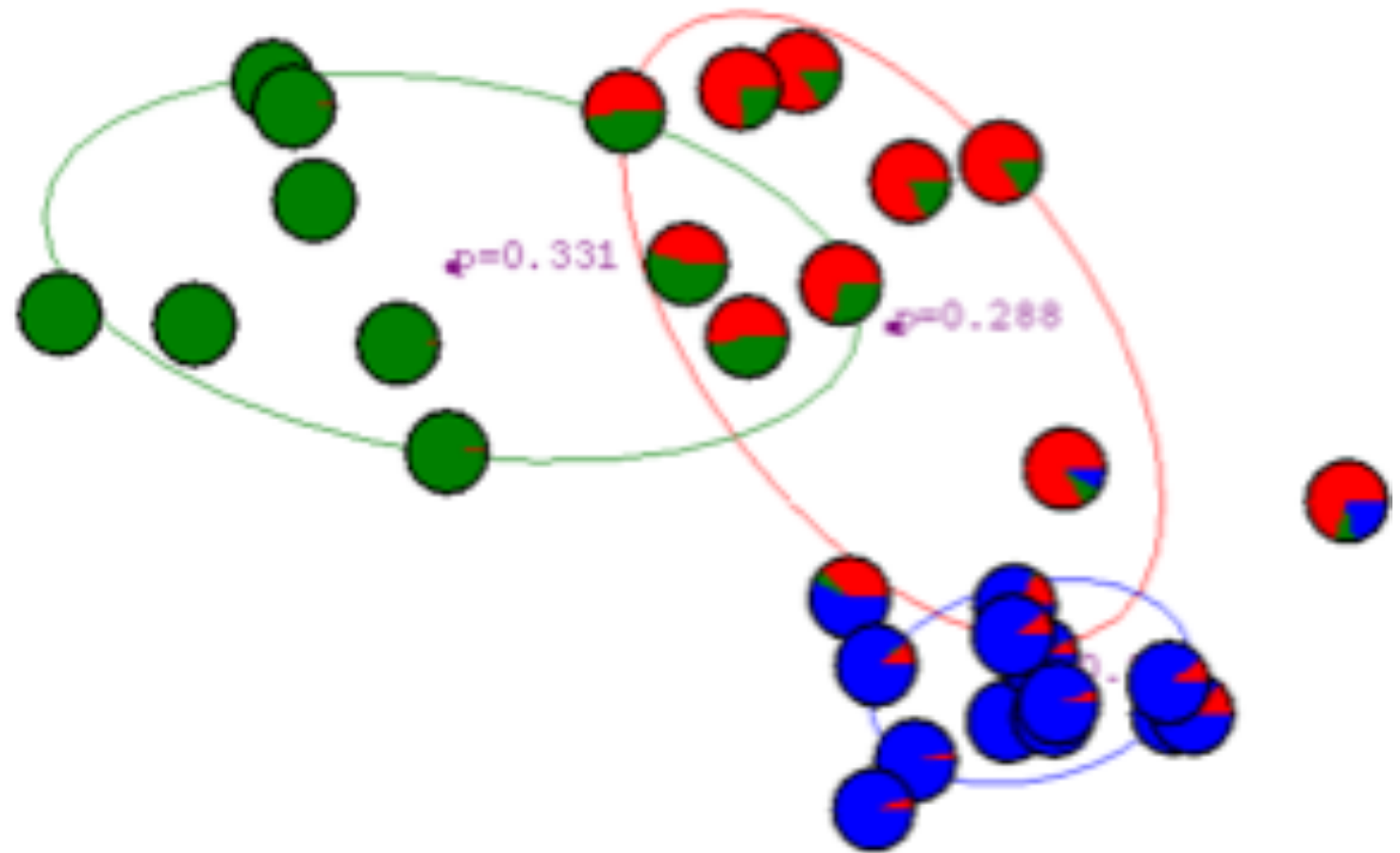
After 2<sup>nd</sup>  
iteration



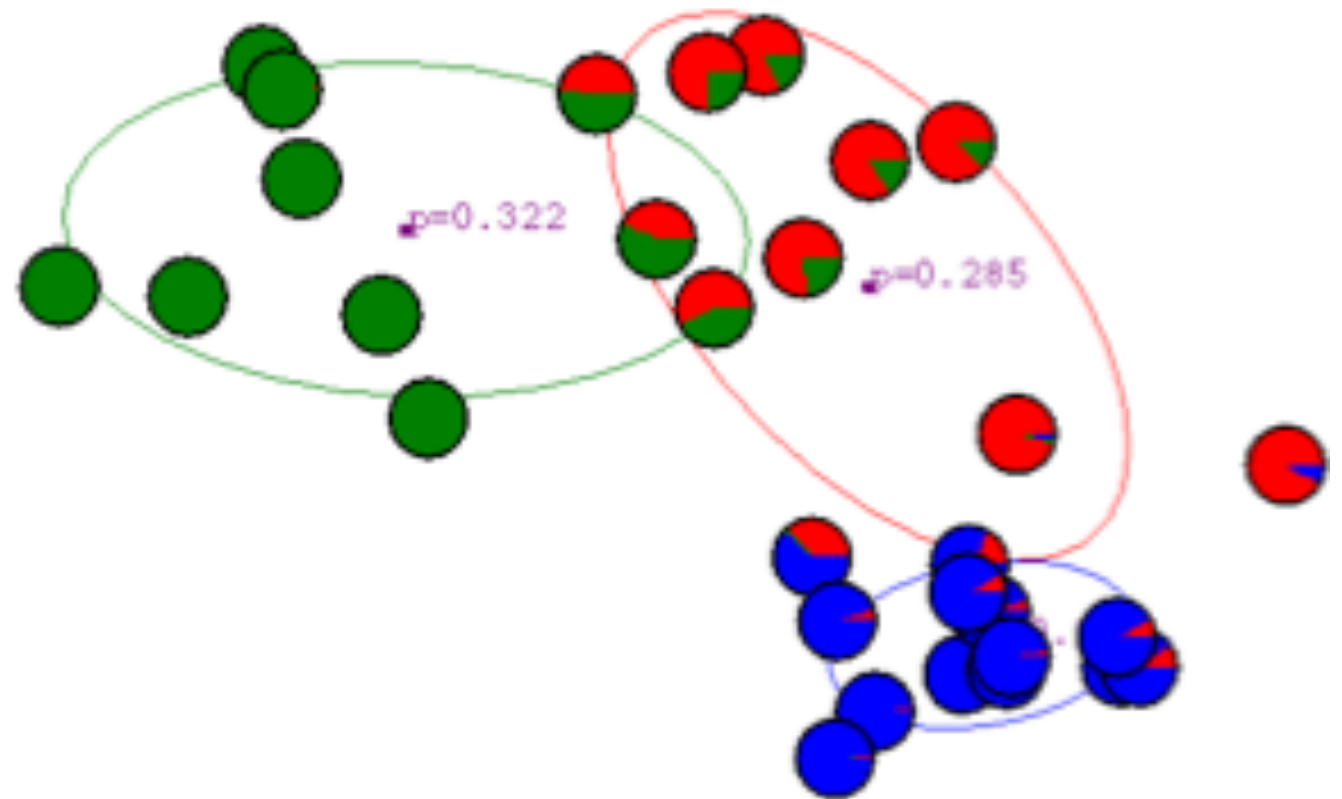
After 3<sup>rd</sup>  
iteration



After 4<sup>th</sup>  
iteration

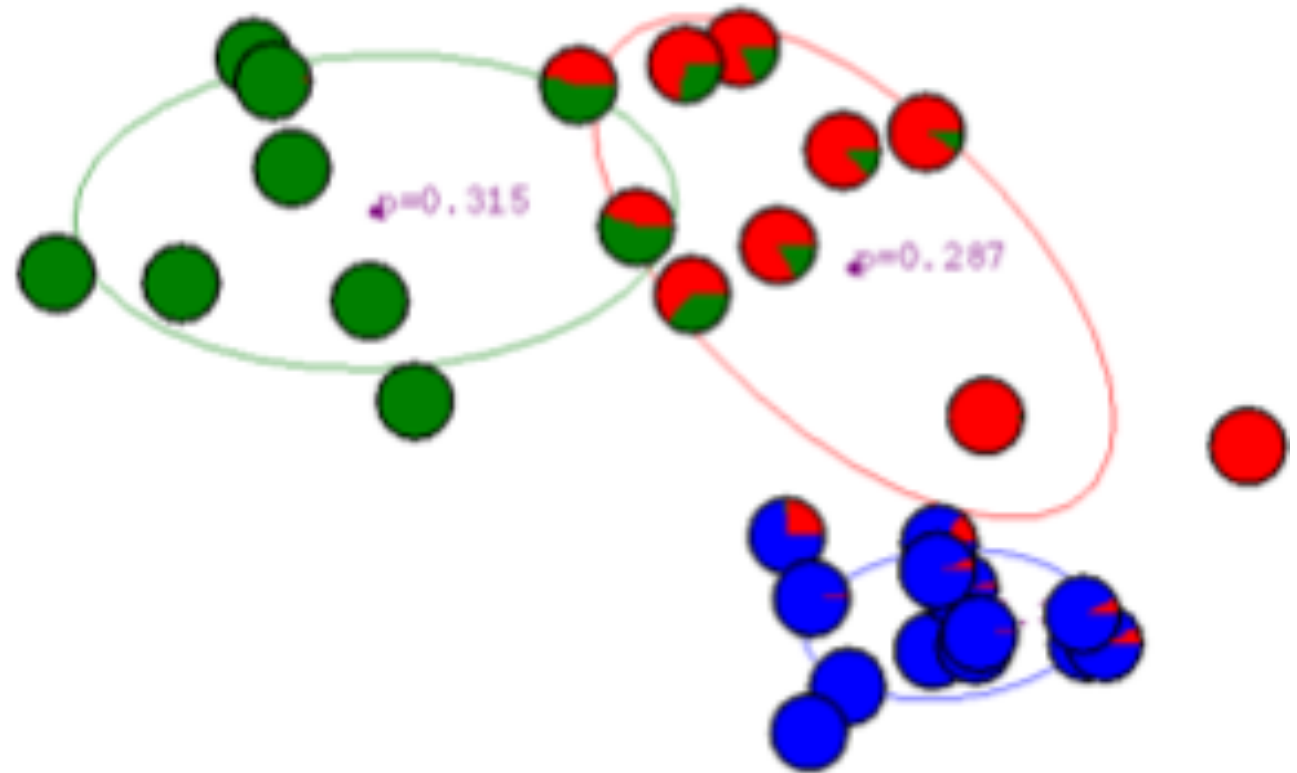


After 5<sup>th</sup>  
iteration

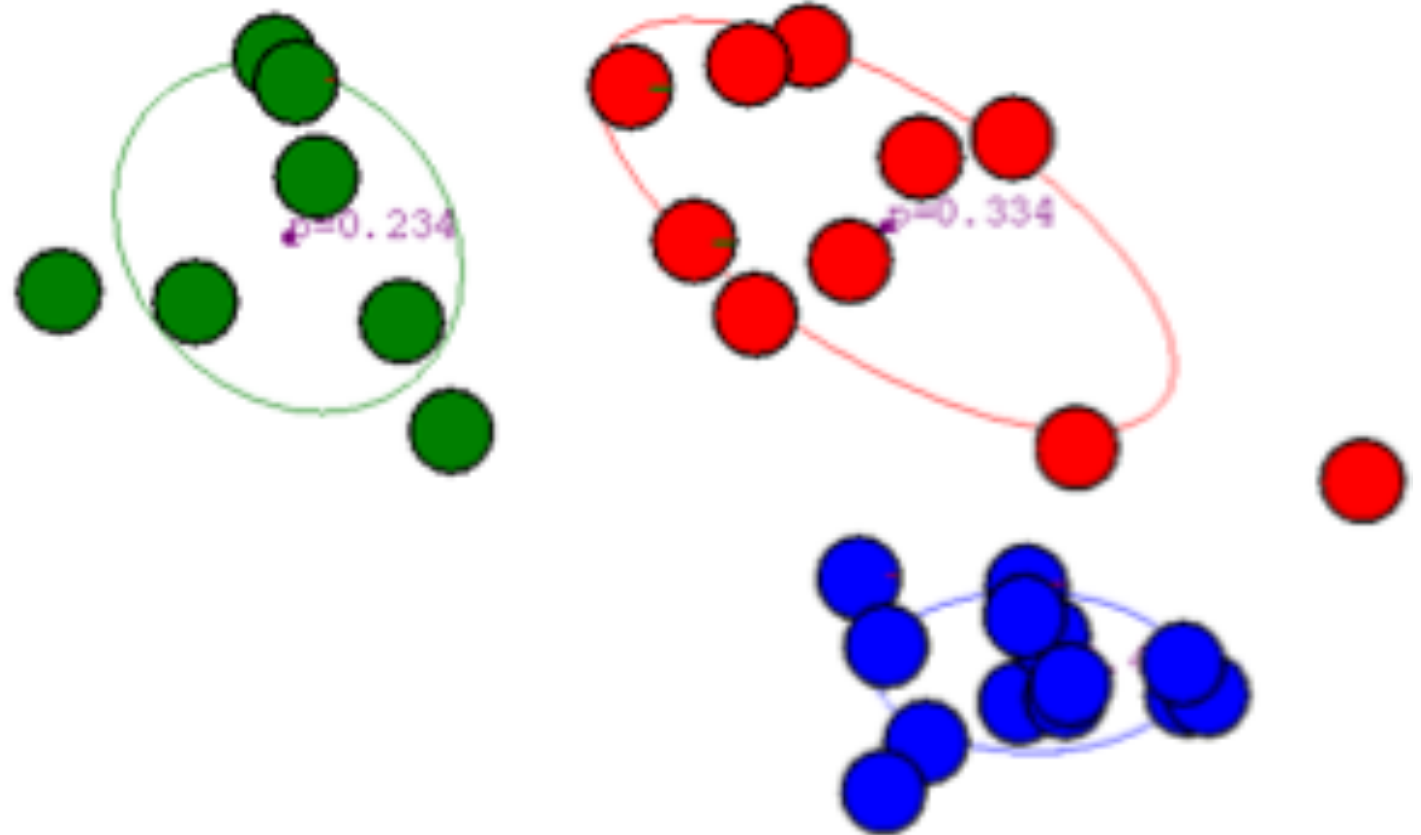




After 6<sup>th</sup>  
iteration



After 20<sup>th</sup>  
iteration



# EM vs Optimisation methods?









## In favour of EM:

- No step size
- Works directly in parameter space model, thus parameter constraints are obeyed
- Fits naturally into graphically model frame work
- Supposedly faster

# Matlab Exercises

```
mu1 = [1 2];  
Sigma1 = [2 0; 0 0.5];  
mu2 = [-3 -5];  
Sigma2 = [1 0; 0 1];  
rng(1); % For reproducibility  
X = [mvnrnd(mu1,Sigma1,1000);mvnrnd(mu2,Sigma2,1000)];  
GMModel = fitgmdist(X,2);  
figure  
y = [zeros(1000,1);ones(1000,1)];  
h = gscatter(X(:,1),X(:,2),y);  
hold on  
ezcontour(@(x1,x2)pdf(GMModel,[x1 x2]),get(gca,  
{'XLim','YLim'}))  
title('{\bf Scatter Plot and Fitted Gaussian Mixture  
Contours}')  
legend(h,'Model 0','Model1')  
hold off
```

**More:** <http://uk.mathworks.com/help/stats/fitgmdist.html#bt8x9hs-2>

	Categorical inputs only	Real-valued inputs only	Mixed Real / Cat okay	Methods
<p>Inputs    Inference Engine    <math>P(E_1 E_2)</math></p>				Joint DE, Bayes Net Structure Learning
<p>Inputs    Classifier    Predict Category</p>	Joint BC Naïve BC	Gauss BC	Dec Tree	Dec Tree, Sigmoid Perceptron, Sigmoid N.Net, Gauss/Joint BC, Gauss Naïve BC, N.Neigh, Bayes Net Based BC, Cascade Correlation, GMM-BC
<p>Inputs    Density Estimator    Probability</p>	Joint DE Naïve DE	Gauss DE		Joint DE, Naïve DE, Gauss/Joint DE, Gauss Naïve DE, Bayes Net Structure Learning, GMMs
<p>Inputs    Regressor    Predict real no.</p>				Linear Regression, Polynomial Regression, Perceptron, Neural Net, N.Neigh, Kernel, LWR, RBFs, Robust Regression, Cascade Correlation, Regression Trees, GMDH, Multilinear Interp, MARS

# Assignment 2

- Repeat Exercise 1.6.1 not using a generated data as shown, but using data you decided for your project.

