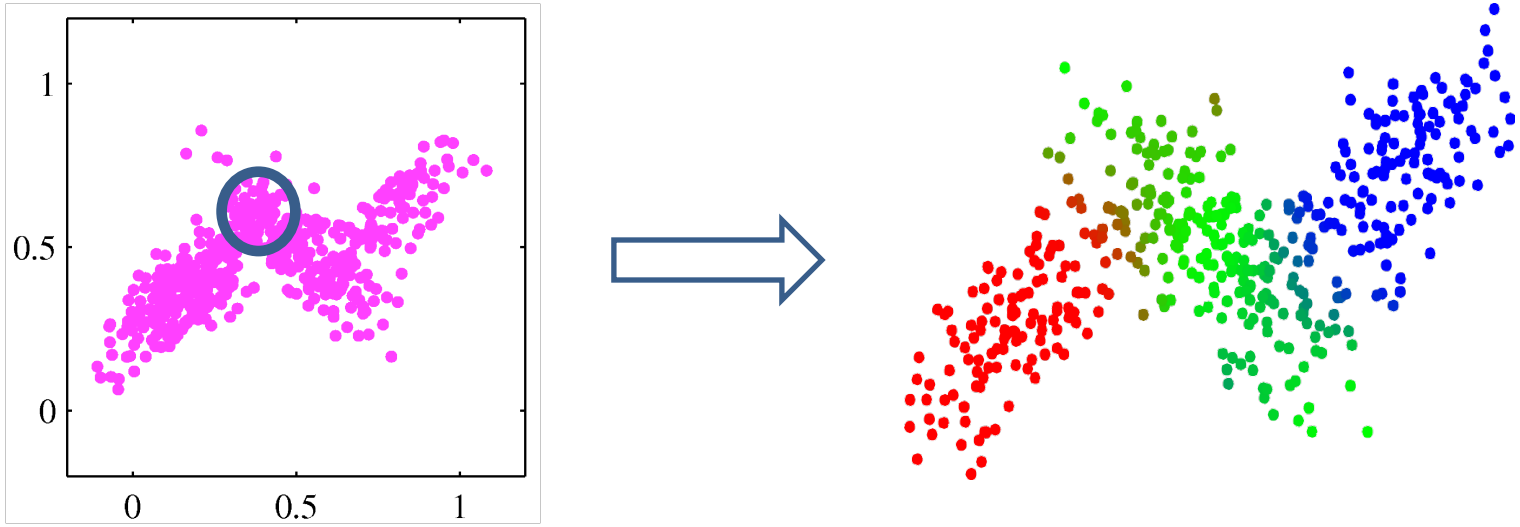
Gaussian Mixture Model (GMM) & Expectation Maximization (EM)

Recap: K-Means

- **Hard assignment**
  - Each data point is assigned to a cluster with 100% probability
  - Clusters may overlap
    - Distance can be deceiving

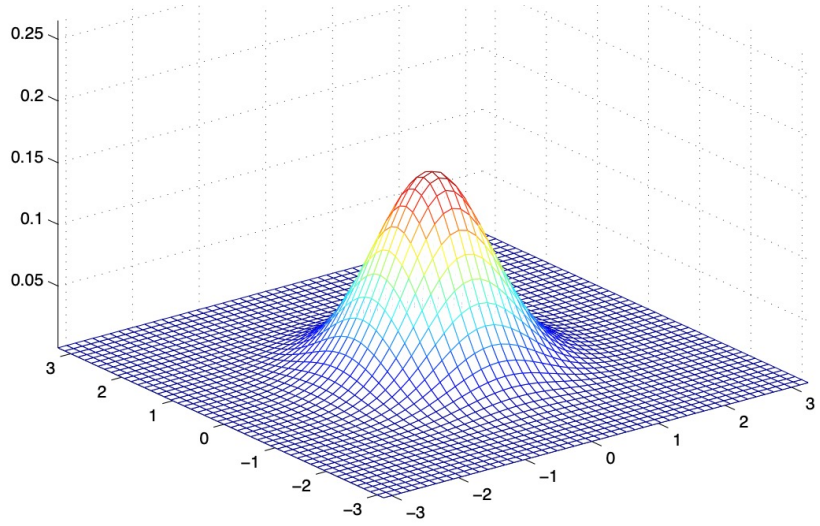


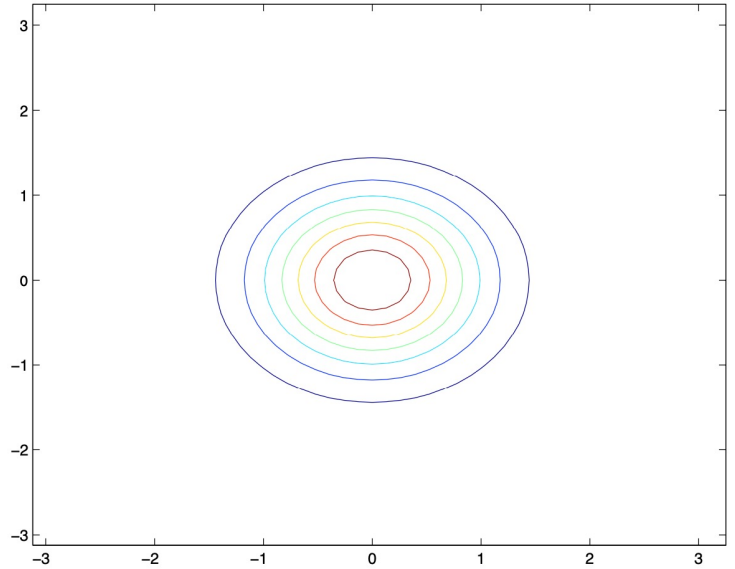
- **Soft assignment**
  - Probability distribution over the cluster assignment
  - E.g., a data point is assigned to cluster 1 with 70% probability, to cluster 2 with 20% probability, and to cluster 3 with 10% probability

Gaussian Mixture Model (GMM)

Multivariate Gaussian Distribution

$$p(\mathbf{x};\mu,\Sigma) = \mathcal{N}(\mathbf{x};\mu,\Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right)$$





$$\mathcal{N}([0;0],\Sigma = \mathbf{I})$$

Mixture Model

- A mixture model is the **linear superposition** (i.e., weighted sum) of a number of probability density functions

$$p(x) = \sum_{k=1}^{K} \pi_k \cdot p_k(x)$$

- The **mixture coefficients** (or weights) are determined by a distribution π

$$\sum_{k=1}^{K} \pi_k = 1, \qquad 0 \leq \pi_k \leq 1$$

- $p_k(x)$ is the density function of k-th distribution

Gaussian Mixture Model (GMM)

- **Gaussian mixture:** A linear superposition of *K* Gaussians

$$p(x) = \sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(x; \mu_k, \Sigma_k)$$

- The **mixture coefficients** π_k satisfy

$$\sum_{k=1}^{K} \pi_k = 1, \qquad 0 \leq \pi_k \leq 1$$

- **Interpretation**
  - The density $N(\mathbf{x}; \mu_k, \Sigma_k)$ is the probability of ***x***, given *k-th* Gaussian was chosen
  - Mixture coefficient π_k *as* prior probabilities of choosing *k-th* Gaussian

GMM: An Example

$$p(x) = \underbrace{0.3}_{\pi_1} \mathcal{N}\left(x \mid \underbrace{\begin{pmatrix} 4 \\ 4.5 \end{pmatrix}}_{\mu_1}, \underbrace{\begin{pmatrix} 1.2 & 0.6 \\ 0.6 & 0.5 \end{pmatrix}}_{\Sigma_1}\right) + \underbrace{0.5}_{\pi_2} \mathcal{N}\left(x \mid \underbrace{\begin{pmatrix} 8 \\ 1 \end{pmatrix}}_{\mu_2}, \underbrace{\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}}_{\Sigma_2}\right) + \underbrace{0.2}_{\pi_3} \mathcal{N}\left(x \mid \underbrace{\begin{pmatrix} 9 \\ 8 \end{pmatrix}}_{\mu_3}, \underbrace{\begin{pmatrix} 0.6 & 0.5 \\ 0.5 & 1.5 \end{pmatrix}}_{\Sigma_3}\right)$$

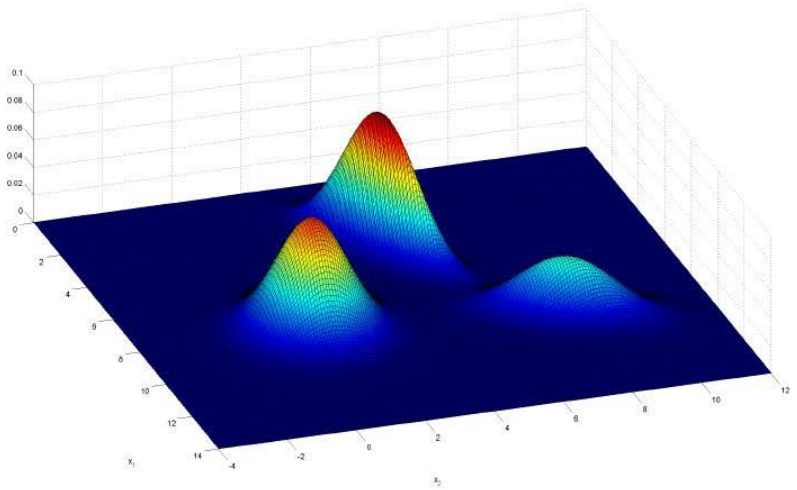


Figure: Probability density function.

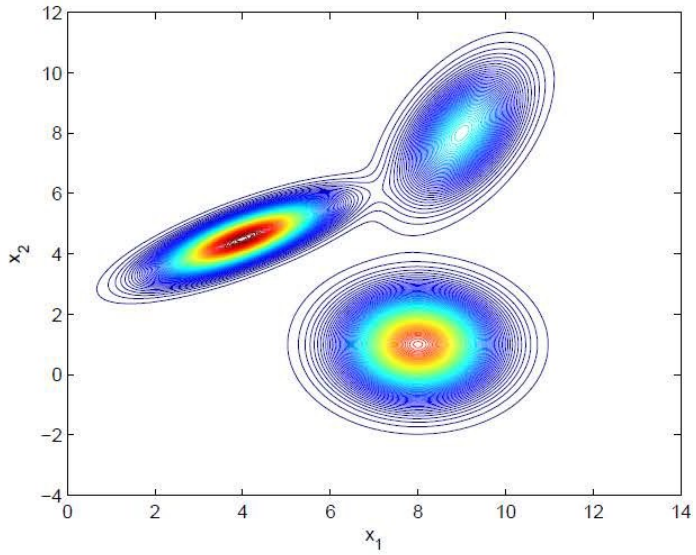


Figure: Contour plot.

Sampling from the GMM

- **Way 1: Per-sample perspective**

  For i=1:N

  Generate a random number based on prior probabilities $\{\pi_k\}$ and determine which of K components to draw a sample from

  Generate a sample from the corresponding Gaussian $N(\mathbf{x}; \mu_k, \Sigma_k)$

  End

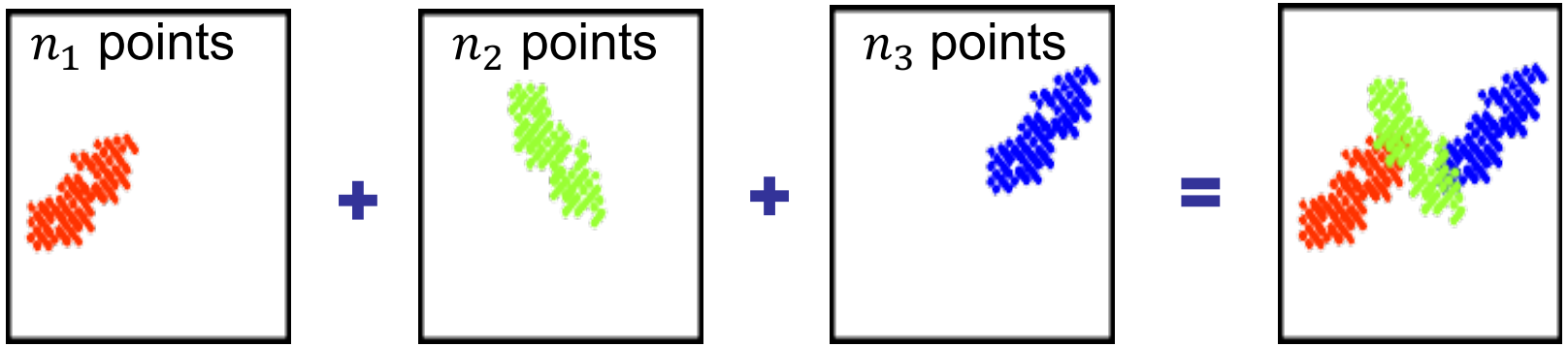
- **Way 2: Gaussian component perspective**

  For k=1:K

  Compute number of samples $n_k = N\,\pi_k$ to draw from the k-th Gaussian

  Generate n_k samples from the k-th Gaussian $N(\mathbf{x}; \mu_k, \Sigma_k)$

  End

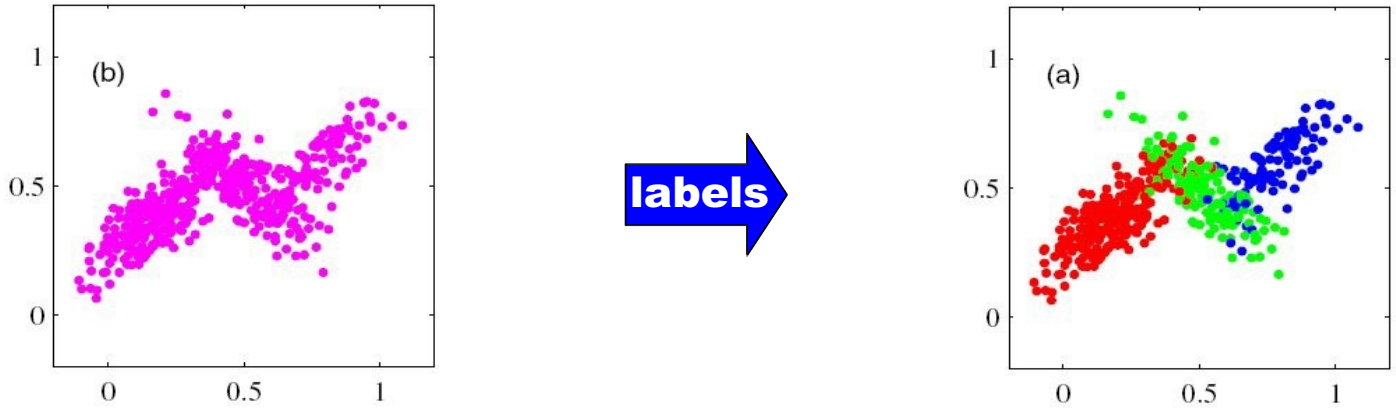


Fitting the GMM

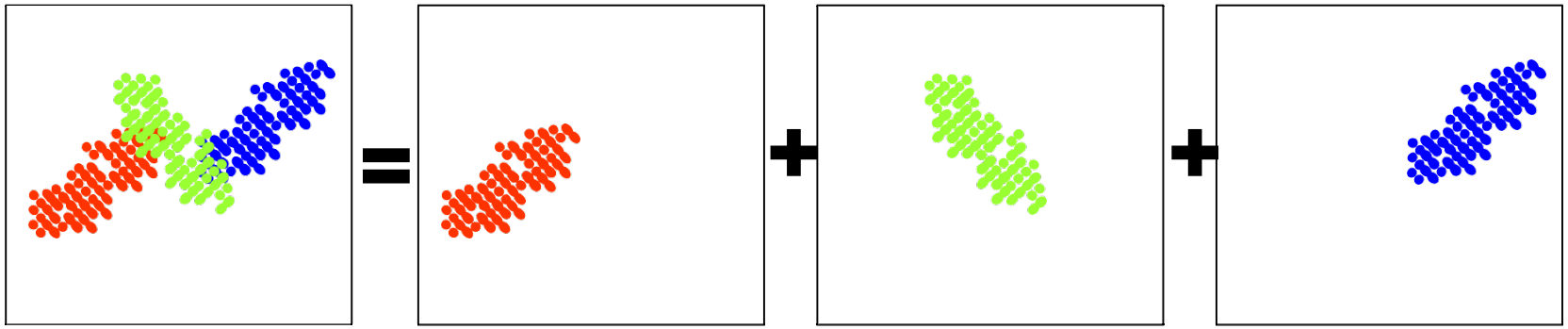
- **Invert the sampling process**
  - Given the unlabeled data set, find the corresponding parameters
    - Mixture coefficients $\{\pi_k\}$
    - Means & covariance matrices $\{\mu_k, \Sigma_k\}$

- If knowing **which Gaussian component** generates each data point
  - Maximum likelihood estimation would fit each component to the corresponding cluster

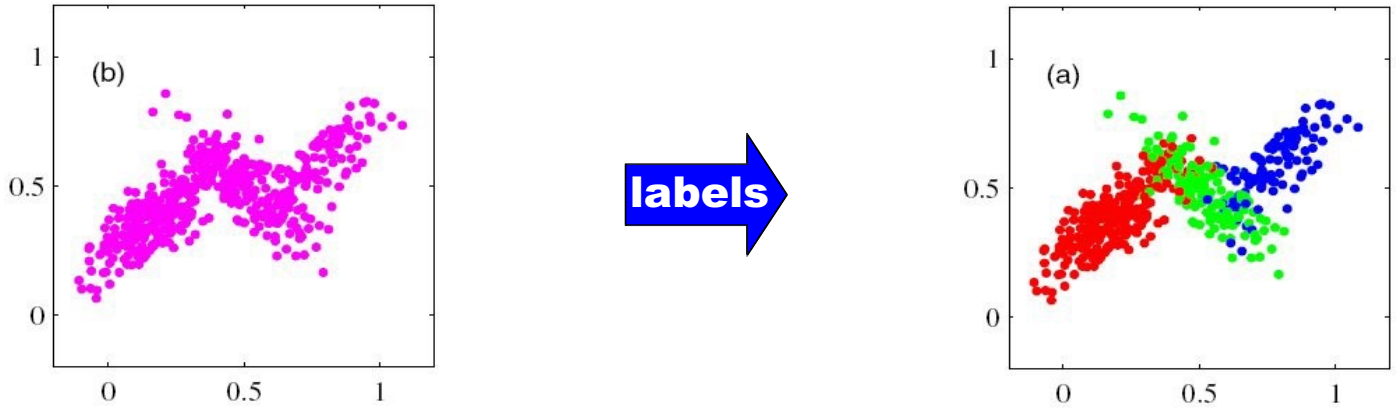
Fitting the GMM with Known Gaussian Components



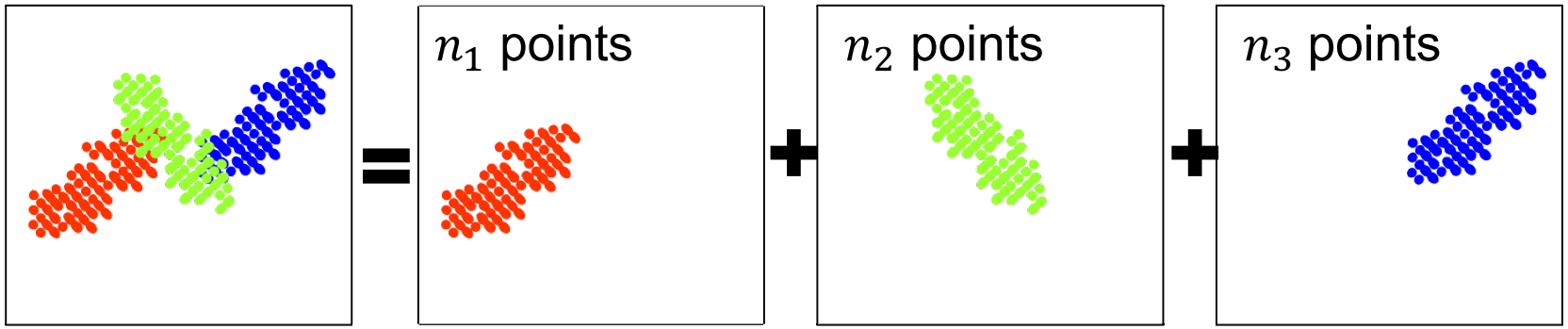
Can recover the underlying generative process, i.e., decomposing components



Fitting the GMM with Known Gaussian Components



Can also easily estimate each Gaussian, along with the mixture weights!



estimate
$\pi_k = n_k / N$

estimate
μ_1, Σ_1

estimate
μ_2, Σ_2

estimate
μ_3, Σ_3

Fitting the GMM

- **Invert the sampling process**
  - Given the unlabeled data set, find the corresponding parameters
    - Mixture coefficients $\{\pi_k\}$
    - Means & covariance matrices $\{\mu_k, \Sigma_k\}$

- If knowing **which Gaussian component** generates each data point
  - Maximum likelihood estimation (MLE) would fit each component to the corresponding cluster

- However, the data set is unlabeled
  - Not knowing which component
  - MLE cannot be directly used to fit the GMM

Log-Likelihood of Unlabeled Data

Given N **unlabeled** data samples $\mathbf{X} = \{\boldsymbol{x}_n\}_{n=1}^{N}$, the log-likelihood is

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln p(\mathbf{x}_n) \qquad p(\mathbf{x}_n) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)$$

$$= \sum_{n=1}^{N} \ln \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)$$

sum over the components inside the *ln*, which is challenging to solve!

We shall refer to the cluster labels as ***latent*** variables

Generative Models for Classification

Data generation process: model distribution of likelihood & label prior

$p(\mathbf{x}|t)$ Likelihood

$p(t)$ Label prior

Bayes rule to derive label t's posterior distribution given x

$$p(t|\mathbf{x}) = \frac{p(\mathbf{x}|t) \cdot p(t)}{p(\mathbf{x})}$$

Classification $\arg\max_t p(t|\mathbf{x}) = \arg\max_t p(\mathbf{x}|t)p(t)$

Clustering: a similar procedure that simulates labels by introducing latent variables

Generative Models for Clustering

Data generation process: model distribution of likelihood & ***latent prior***

$p(\mathbf{x}|\mathbf{z})$ Likelihood $\qquad$ $p(\mathbf{z})$ Latent prior

Bayes rule to derive **latent z**'s posterior distribution given x

$$p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{x}|\mathbf{z}) \cdot p(\mathbf{z})}{p(\mathbf{x})}$$

Clustering (hard assignment) $\qquad \arg\max_k p(z_k|\mathbf{x})$

Generative Process of GMM

- With probability 0.7, choose **component 1**, otherwise choose component 2

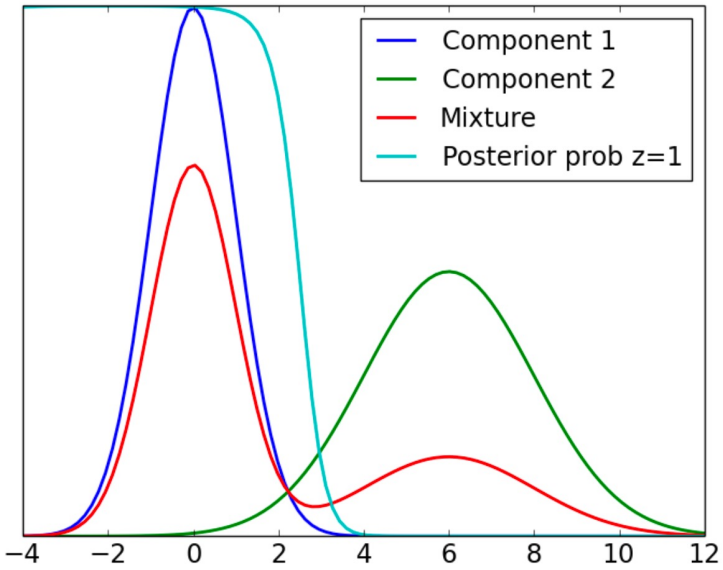
$$\mathbf{z} \sim \text{Multi}(0.7, 0.3)$$

- If we chose component 1, then sample x from a Gaussian with mean 0 and standard deviation 1

$$p(x|z_1 = 1) \sim \mathcal{N}(x; 0, 1)$$

- If we chose component 2, then sample x from a Gaussian with mean 6 and standard deviation 2

$$p(x|z_2 = 1) \sim \mathcal{N}(x; 6, 2)$$



$$p(x) = 0.7 \cdot \mathcal{N}(x; 0, 1) + 0.3 \cdot \mathcal{N}(x; 6, 2)$$

General case

$$\mathbf{z} \sim \text{Multi}(\pi_1, \cdots, \pi_K)$$

$$p(\mathbf{x}|z_k = 1) \sim \mathcal{N}(\mathbf{x}; \mu_k, \Sigma_k)$$

GMM: Latent Variable View

Let us introduce a K-dim binary random variable **z** having 1-of-K encodings

z's nonzero element indicates which of the K Gaussians that x comes from

- z_k =1 implies that x is from the k-th Gaussian

$$p(\mathbf{x}|z_k = 1) = \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k) \quad \Rightarrow \quad p(\mathbf{x}|\mathbf{z}) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)^{z_k}$$

Mixture coefficient π_k as prior probabilities of choosing *k-th* Gaussian

$$p(z_k = 1) = \pi_k \quad \Rightarrow \quad p(\mathbf{z}) = \pi_1^{z_1}\pi_2^{z_2}\cdots\pi_K^{z_K} = \prod_{k=1}^{K} \pi_k^{z_k}$$

Homework problem

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{z})p(\mathbf{x}|\mathbf{z}) \quad \Rightarrow \quad p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}; \mu_k, \Sigma_k)$$

Latent variable view for GMM

Log-Likelihood with Latent Variables

Given the ***joint*** data samples and latent variables $(\mathrm{X}, Z) = \{\boldsymbol{x}_n, \boldsymbol{z}_n\}_{n=1}^{N}$, the log-likelihood is

$$p(\mathbf{X}, \mathbf{Z}|\pi, \mu, \Sigma) = \prod_{n=1}^{N} p(\mathbf{x}_n, \mathbf{z}_n) = \prod_{n=1}^{N} p(\mathbf{z}_n) p(\mathbf{x}_n|\mathbf{z}_n)$$

$$= \prod_{n=1}^{N} \prod_{k=1}^{K} \pi_k^{z_{nk}} \mathcal{N}(x_n|\mu_k, \Sigma_k)^{z_{nk}}$$

$$\ln p(\mathbf{X}, \mathbf{Z}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \{\ln \pi_k + \ln \mathcal{N}(x_n|\mu_k, \Sigma_k)\}$$

NO sum over the components appears inside the ln, which is easy to solve!

Log-Likelihood: Summary

$$\ln p(\mathbf{X}, \mathbf{Z}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \{\ln \pi_k + \ln \mathcal{N}(x_n|\mu_k, \Sigma_k)\}$$

Likelihood

$$p(\mathbf{x}_n|\mathbf{z}_n) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x_n}|\mu_k, \Sigma_k)^{z_{nk}}$$

Latent prior

$$p(\mathbf{z}_n) = \prod_{k=1}^{K} \pi_k^{z_{nk}}$$

Clustering
(hard assignment)

$$\arg\max_k p(z_k|\mathbf{x}_n) = \frac{p(z_{nk}=1)p(\mathbf{x}_n|z_{nk}=1)}{\sum_{k=1}^{K} p(z_{nk}=1)p(\mathbf{x}_n|z_{nk}=1)}$$

$$= \frac{\pi_k \mathcal{N}(\mathbf{x_n}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x_n}|\mu_j, \Sigma_j)}$$

Goal

Learn model parameters $\{\pi_k, \mu_k, \Sigma_k\}$

An elegant and powerful method for **finding maximum log-likelihood** solutions for models with **latent variables** is called the **expectation-maximization (EM)** algorithm

Expectation Maximization (EM)

Hard Expectation Maximization (EM)

Initialize model **parameters** **randomly**

while not converged

- **E-Step: guess the value of latent variables**
  - Set **latent variables** to the values that maximizes the likelihood

    (treating parameters as observed)

- **M-Step: update the parameters of the model based on the guess**
  - Set the **parameters** to the values that maximizes the likelihood

    (treating latent variables as observed)

Hard EM for GMM with Hard Assignment

Algorithm 1 Hard EM for GMMs

procedure HARDEM($\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$)

Randomly initialize parameters, $\boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$

while not converged **do**

E-Step:

$$z^{(i)} \leftarrow \operatorname*{argmax}_z \log p(\mathbf{x}^{(i)}|z; \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \log p(z; \boldsymbol{\phi})$$

Each date point is assigned to a single cluster

M-Step:

Similar to K-means

$$\phi_k \leftarrow \frac{1}{N} \sum_{i=1}^N \mathbb{I}(z^{(i)} = k), \forall k$$

$$\boldsymbol{\mu}_k \leftarrow \frac{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)\mathbf{x}^{(i)}}{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)}, \forall k$$

$$\boldsymbol{\Sigma}_k \leftarrow \frac{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)^T}{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)}, \forall k$$

return $(\boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$

GMM (Hard Assignment) vs. Gaussian Discriminant Analysis

Data: $\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ where $\mathbf{x}^{(i)} \in \mathbb{R}^M$

Generative process:

$$z \sim \text{Categorical}(\boldsymbol{\phi})$$
$$\mathbf{x} \sim \text{Gaussian}(\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z)$$

Model:

Joint: $p(\mathbf{x}, z; \boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = p(\mathbf{x}|z; \boldsymbol{\mu}, \boldsymbol{\Sigma})p(z; \boldsymbol{\phi})$

Marginal: $p(\mathbf{x}; \boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{z=1}^K p(\mathbf{x}|z; \boldsymbol{\mu}, \boldsymbol{\Sigma})p(z; \boldsymbol{\phi})$

(Marginal) Log-likelihood:

$$\ell(\boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^N p(\mathbf{x}^{(i)}; \boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$
$$= \sum_{i=1}^N \log \sum_{z=1}^K p(\mathbf{x}^{(i)}|z; \boldsymbol{\mu}, \boldsymbol{\Sigma})p(z; \boldsymbol{\phi})$$

Data: $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{z}^{(i)})\}_{i=1}^N$ where $\mathbf{x}^{(i)} \in \mathbb{R}^M$ and $z^{(i)} \in \{1, \ldots, K\}$

Generative process:

$$z \sim \text{Categorical}(\boldsymbol{\phi})$$
$$\mathbf{x} \sim \text{Gaussian}(\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z)$$

Model:

Joint: $p(\mathbf{x}, z; \boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = p(\mathbf{x}|z; \boldsymbol{\mu}, \boldsymbol{\Sigma})p(z; \boldsymbol{\phi})$

Log-likelihood:

$$\ell(\boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^N p(\mathbf{x}^{(i)}, z^{(i)}; \boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$
$$= \sum_{i=1}^N \log p(\mathbf{x}^{(i)}|z^{(i)}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \log p(z^{(i)}; \boldsymbol{\phi})$$

GDA: Maximum Likelihood Estimation

Data: $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{z}^{(i)})\}_{i=1}^{N}$ where $\mathbf{x}^{(i)} \in \mathbb{R}^M$ and $z^{(i)} \in \{1, \ldots, K\}$

Log-likelihood:

$$\ell(\boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^{N} \log p(\mathbf{x}^{(i)} | z^{(i)}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \log p(z^{(i)}; \boldsymbol{\phi})$$

Maximum Likelihood Estimates:

$$\phi_k = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(z^{(i)} = k), \forall k$$

$$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^{N} \mathbb{I}(z^{(i)} = k) \mathbf{x}^{(i)}}{\sum_{i=1}^{N} \mathbb{I}(z^{(i)} = k)}, \forall k$$

$$\boldsymbol{\Sigma}_k = \frac{\sum_{i=1}^{N} \mathbb{I}(z^{(i)} = k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)^T}{\sum_{i=1}^{N} \mathbb{I}(z^{(i)} = k)}, \forall k$$

Implementation: Just counting

Hard EM for GMMs

Algorithm 1 Hard EM for GMMs

procedure HARDEM($\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$)

Randomly initialize parameters, $\boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$

while not converged **do**

E-Step:

$$z^{(i)} \leftarrow \operatorname*{argmax}_z \log p(\mathbf{x}^{(i)}|z; \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \log p(z; \boldsymbol{\phi})$$

Implementation: For loop over possible values of latent variable

M-Step:

$$\phi_k \leftarrow \frac{1}{N} \sum_{i=1}^N \mathbb{I}(z^{(i)} = k), \forall k$$

$$\boldsymbol{\mu}_k \leftarrow \frac{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)\mathbf{x}^{(i)}}{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)}, \forall k$$

$$\boldsymbol{\Sigma}_k \leftarrow \frac{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)^T}{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)}, \forall k$$

Implementation: Exactly the same as supervised GDA

return $(\boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$

K-Means via Hard EM

Repeat

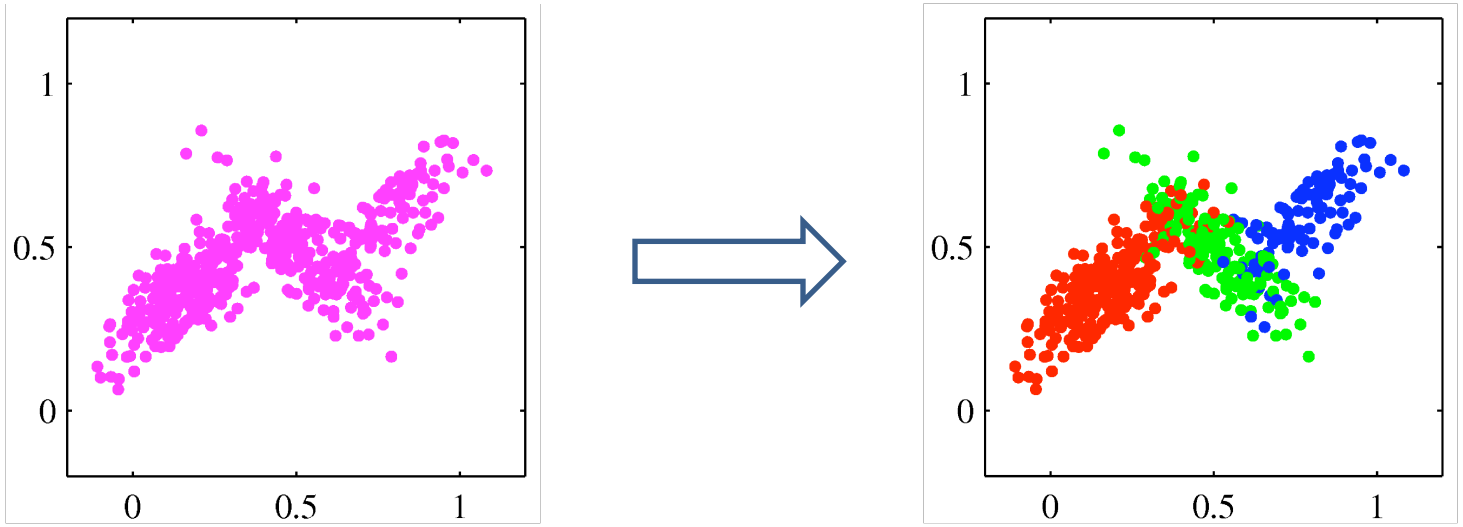
- **E Step**: For given cluster centroids, find optimal cluster assignments

$$r_{ik} = \mathbf{1}[k = \arg\min_j \|\mathbf{x}_i - \mu_j\|_2^2]$$

- **M Step**: For given cluster assignments, find optimal cluster centroids

$$\mu_k = \frac{\sum_i r_{ik}\mathbf{x}_i}{\sum_i r_{ik}}$$

GMM with Hard Assignment



Each data point is assigned to a cluster with 100% probability

Hard Expectation Maximization

Initialize model **parameters** **randomly**

while not converged

- **E-Step: guess the value of latent variables**
  - Set **latent variables** to the values that maximizes the likelihood

    (treating parameters as observed)

- **M-Step: update the parameters of the model based on the guess**
  - Set the **parameters** to the values that maximizes the likelihood

    (treating latent variables as observed)

Soft Expectation Maximization: Soft Assignment

Initialize model **parameters** **randomly**

while not converged

- **E-Step: guess the posterior probabilities of latent variables**
  - Obtain posterior probabilities of the **latent variables**
  - Use **posterior probabilities** as the probability to assign each cluster (treating parameters as observed)

$$p(z_k|\mathbf{x}_n) = \frac{\pi_k \mathcal{N}(\mathbf{x_n}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x_n}|\mu_j, \Sigma_j)}$$

Hard EM: $r_{nk} = \arg\max_k p(z_k|\mathbf{x}_n)$

Soft EM: $\mathbf{c}_n = [p(z_1|\mathbf{x}_n), \cdots, p(z_K|\mathbf{x}_n)]$ Probability distribution over the cluster assignment

Soft Expectation Maximization: Soft Assignment

Initialize model **parameters** **randomly**

while not converged

- **E-Step: guess the posterior probabilities of latent variables**
  - Obtain posterior probabilities of the **latent variables**
  - Use **posterior probabilities** as the probability to assign each cluster (treating parameters as observed)

- **M-Step: update the parameters of the model based on the guess**

  Set the **parameters** to the values that maximizes the likelihood

  (treating latent variables as observed)

Insights of Soft Assignment

- By replacing the ***binary*** latent variables with their ***continuous*** expected values
  - All data points contribute to the estimation of ***all*** components
- Each data point has unit mass to contribute, but splits it across the K components
- The weight of a data point contributes to a component is proportional to the posterior probability that the data point was generated by that component

Hard EM vs. Soft EM

Algorithm 1 Hard EM for GMMs

procedure HARDEM($\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$)

  Randomly initialize parameters, $\phi, \boldsymbol{\mu}, \boldsymbol{\Sigma}$

  while not converged **do**

    E-Step:

$$z^{(i)} \leftarrow \operatorname*{argmax}_z \log p(\mathbf{x}^{(i)}|z; \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \log p(z; \boldsymbol{\phi})$$

    M-Step:

$$\phi_k \leftarrow \frac{1}{N} \sum_{i=1}^N \mathbb{I}(z^{(i)} = k), \forall k$$

$$\boldsymbol{\mu}_k \leftarrow \frac{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)\mathbf{x}^{(i)}}{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)}, \forall k$$

$$\boldsymbol{\Sigma}_k \leftarrow \frac{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)^T}{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)}, \forall k$$

Implementation: Just counting as in supervised GDA

  return $(\phi, \boldsymbol{\mu}, \boldsymbol{\Sigma})$

Algorithm 1 Soft EM for GMMs

procedure SOFTEM($\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$)

  Randomly initialize parameters, $\phi, \boldsymbol{\mu}, \boldsymbol{\Sigma}$

  while not converged **do**

    E-Step:

$$c_k^{(i)} \leftarrow p(z^{(i)} = k|\mathbf{x}^{(i)}; \boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

    M-Step:

$$\phi_k \leftarrow \frac{1}{N} \sum_{i=1}^N c_k^{(i)} \forall k$$

$$\boldsymbol{\mu}_k \leftarrow \frac{\sum_{i=1}^N c_k^{(i)}\mathbf{x}^{(i)}}{\sum_{i=1}^N c_k^{(i)}}, \forall k$$

$$\boldsymbol{\Sigma}_k \leftarrow \frac{\sum_{i=1}^N c_k^{(i)}(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)^T}{\sum_{i=1}^N c_k^{(i)}}, \forall k$$

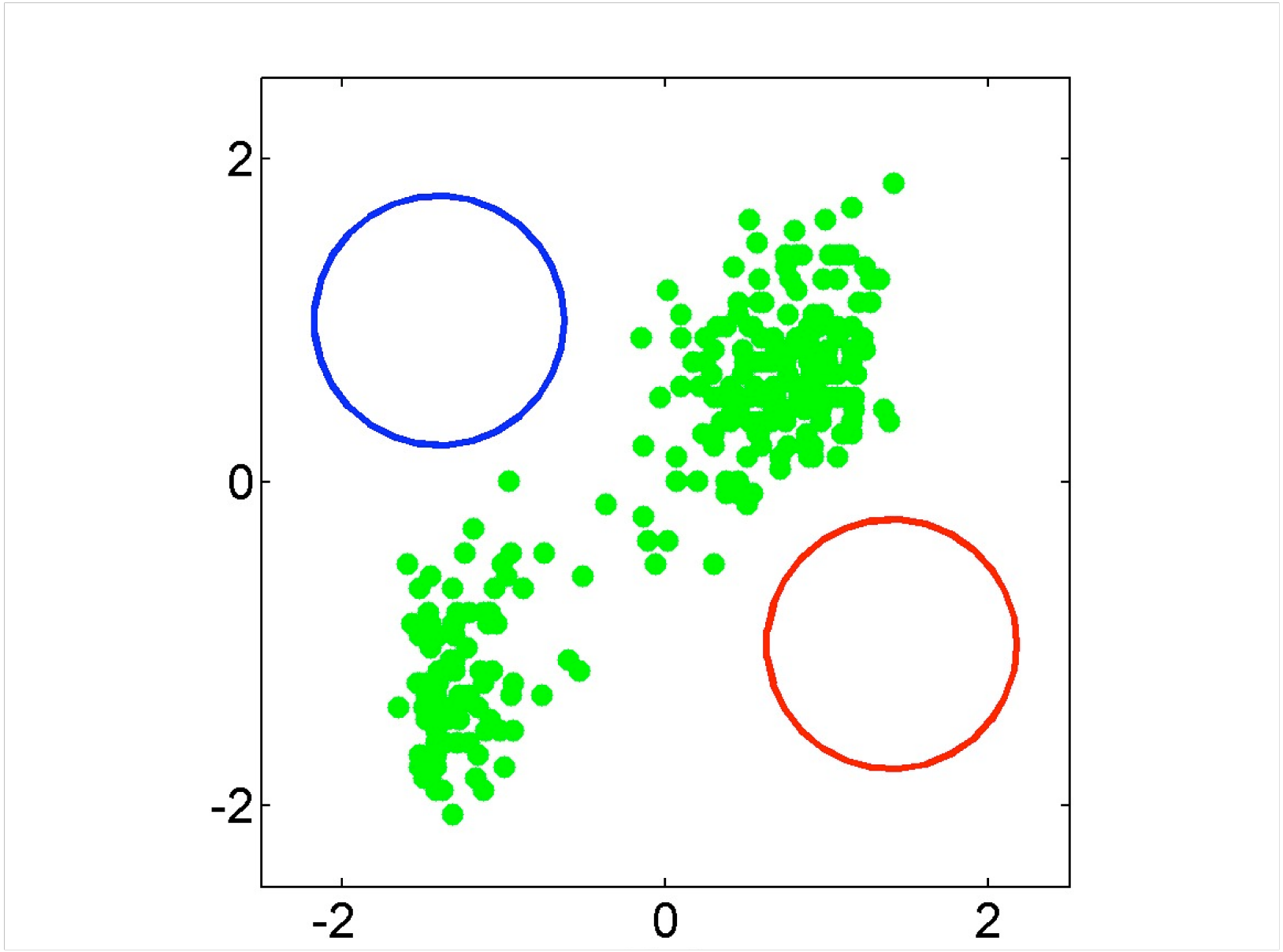
Implementation: Calculation using posterior probabilities

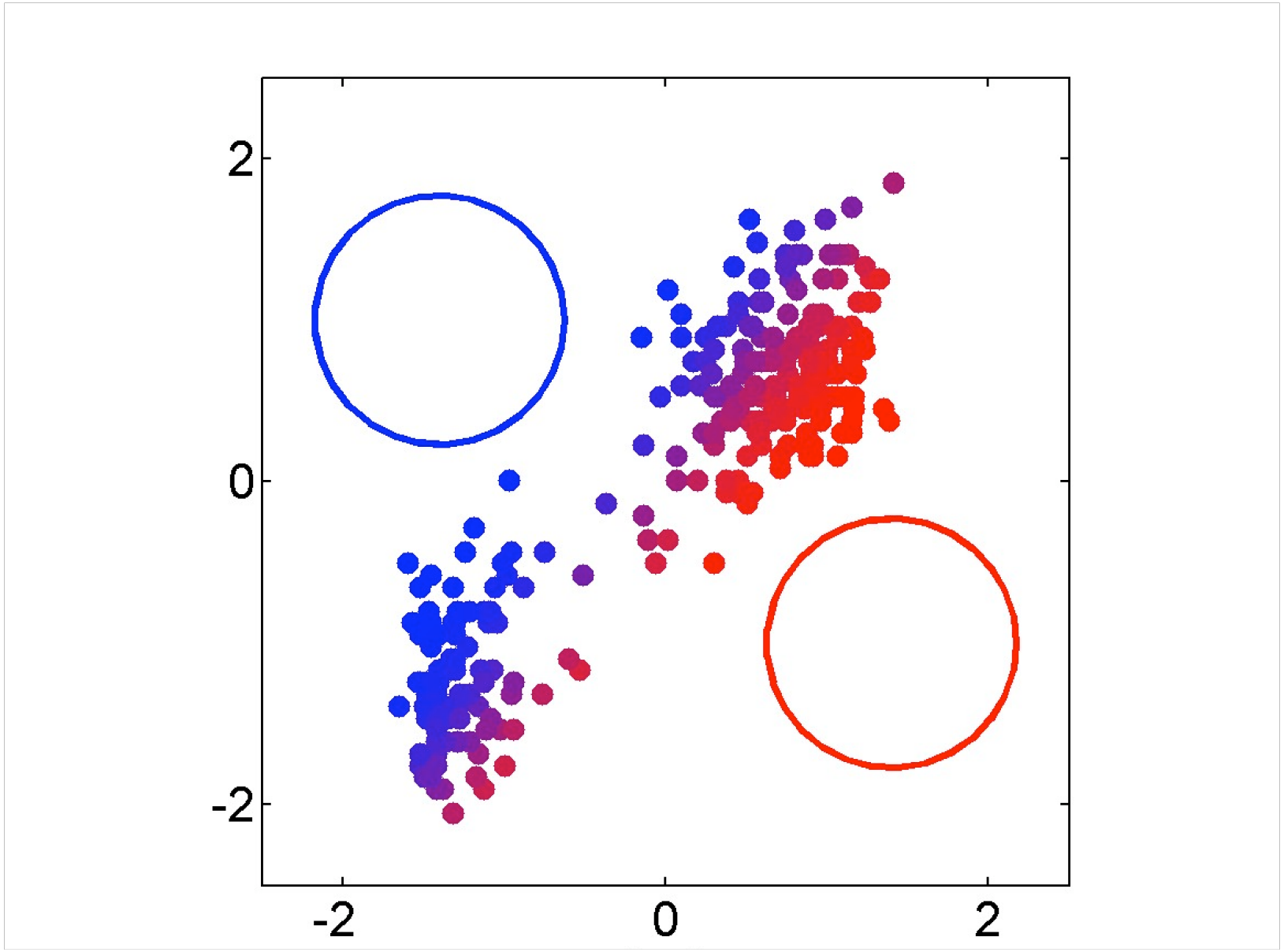
  return $(\phi, \boldsymbol{\mu}, \boldsymbol{\Sigma})$

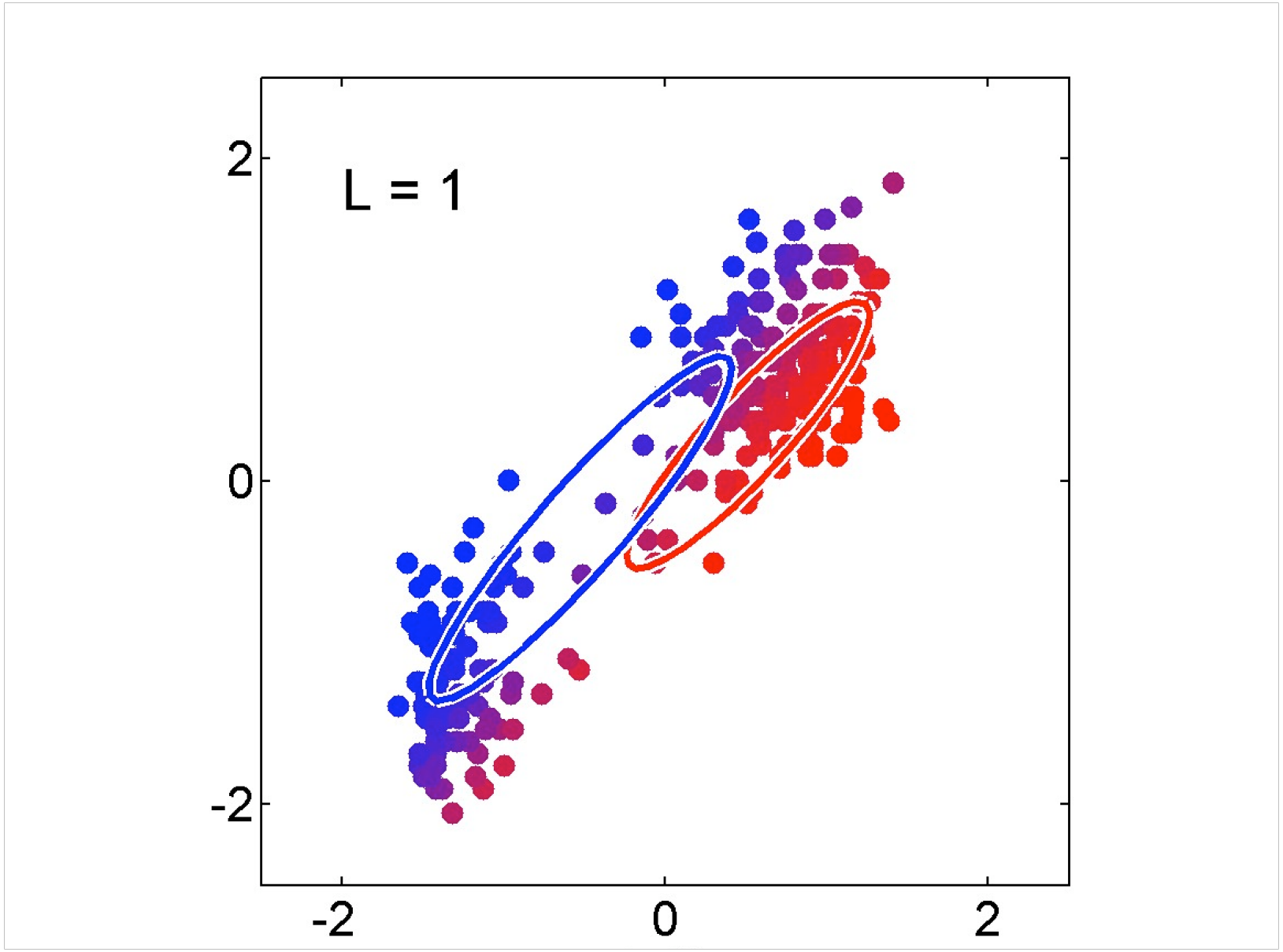
Properties of EM

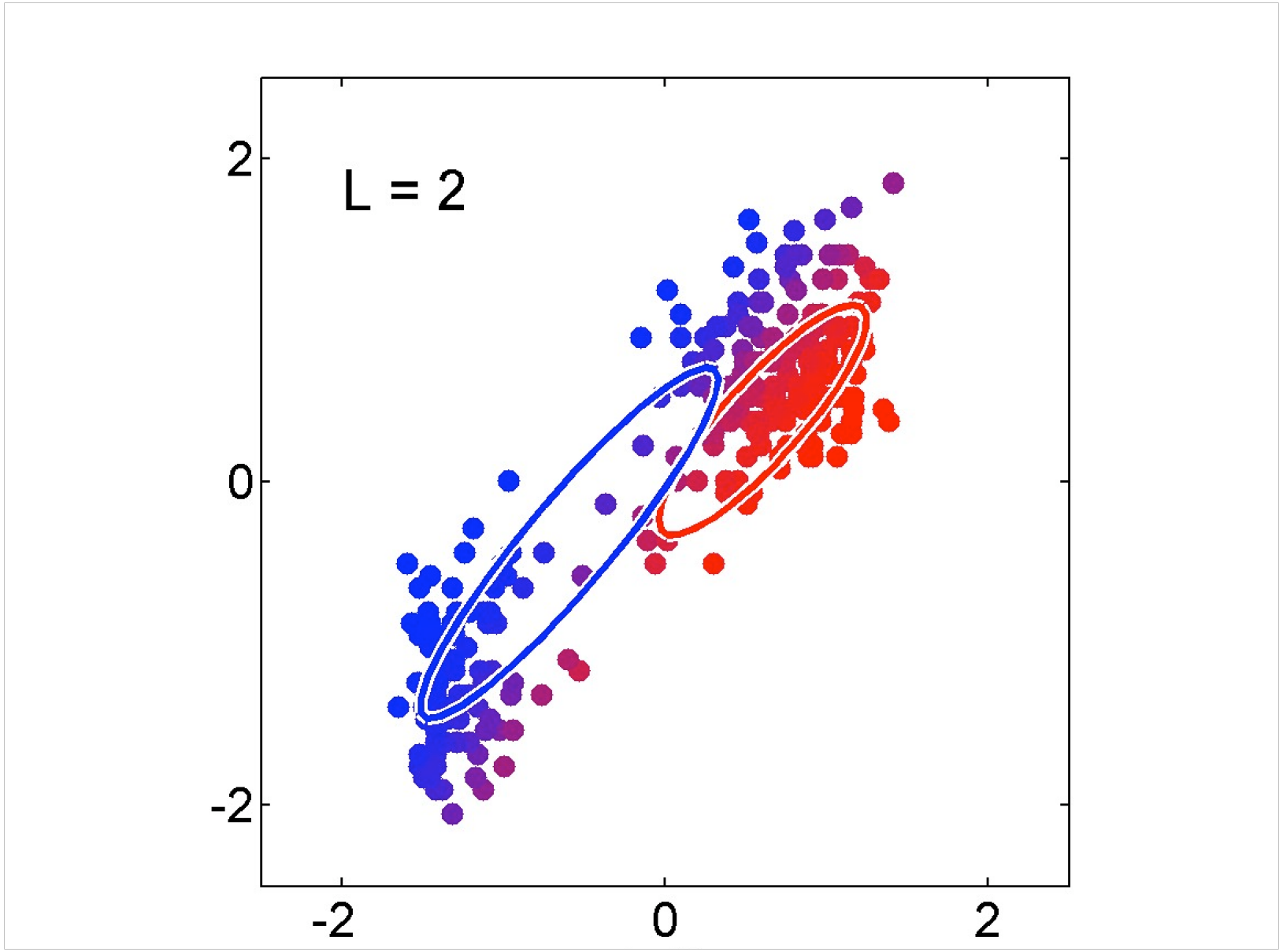
- EM is *trying* to optimize a **nonconvex** function
- But EM is a **local** optimization algorithm
  - EM converges to a local minima
  - E-Step and M-Step can never decrease likelihood
- Typical solution: **Random Restarts**
  - Just like K-Means, we run the algorithm many times
  - Each time initialize parameters randomly
  - Pick the model parameters that give the highest likelihood

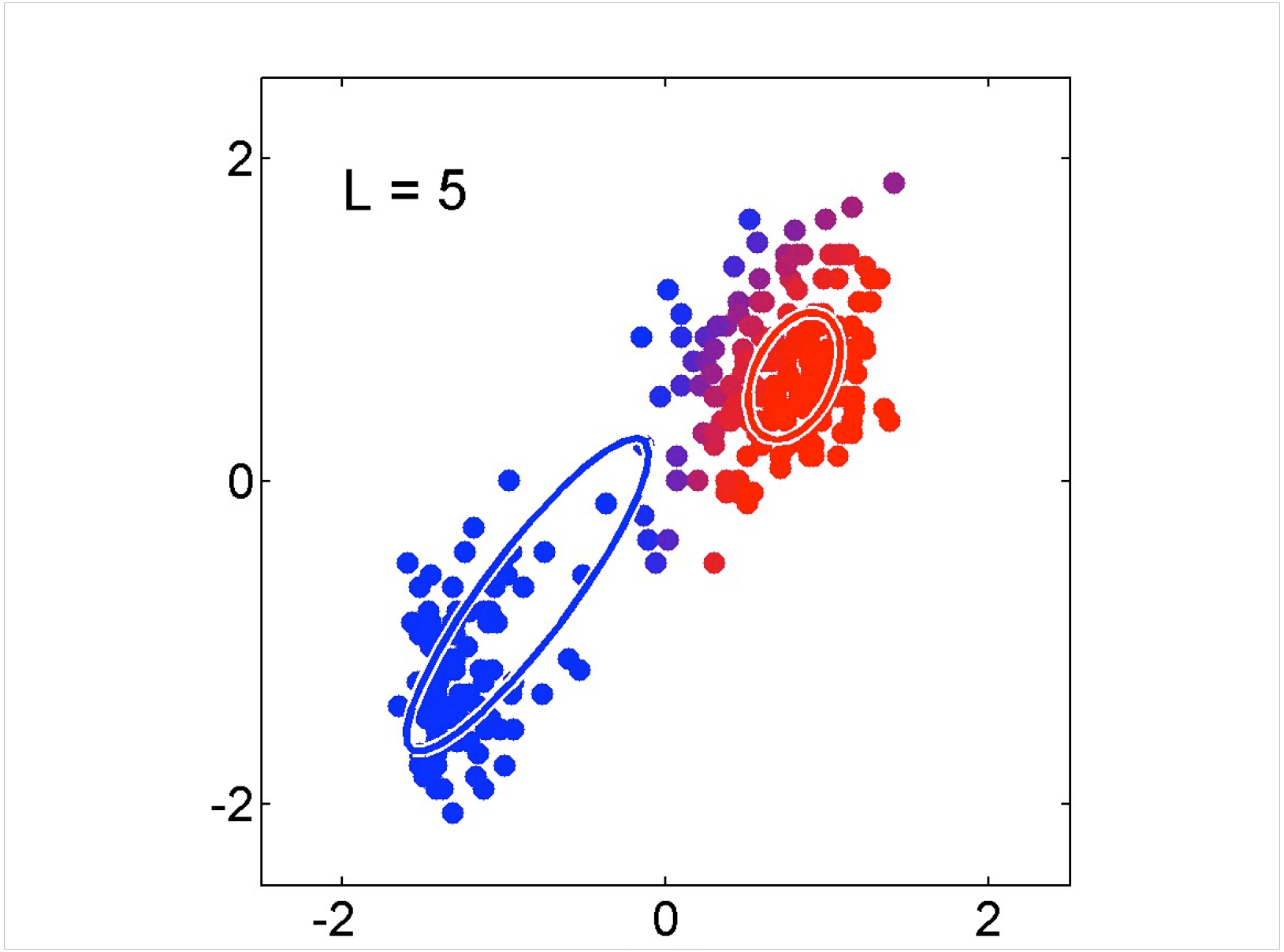
Soft EM for GMM: An Example

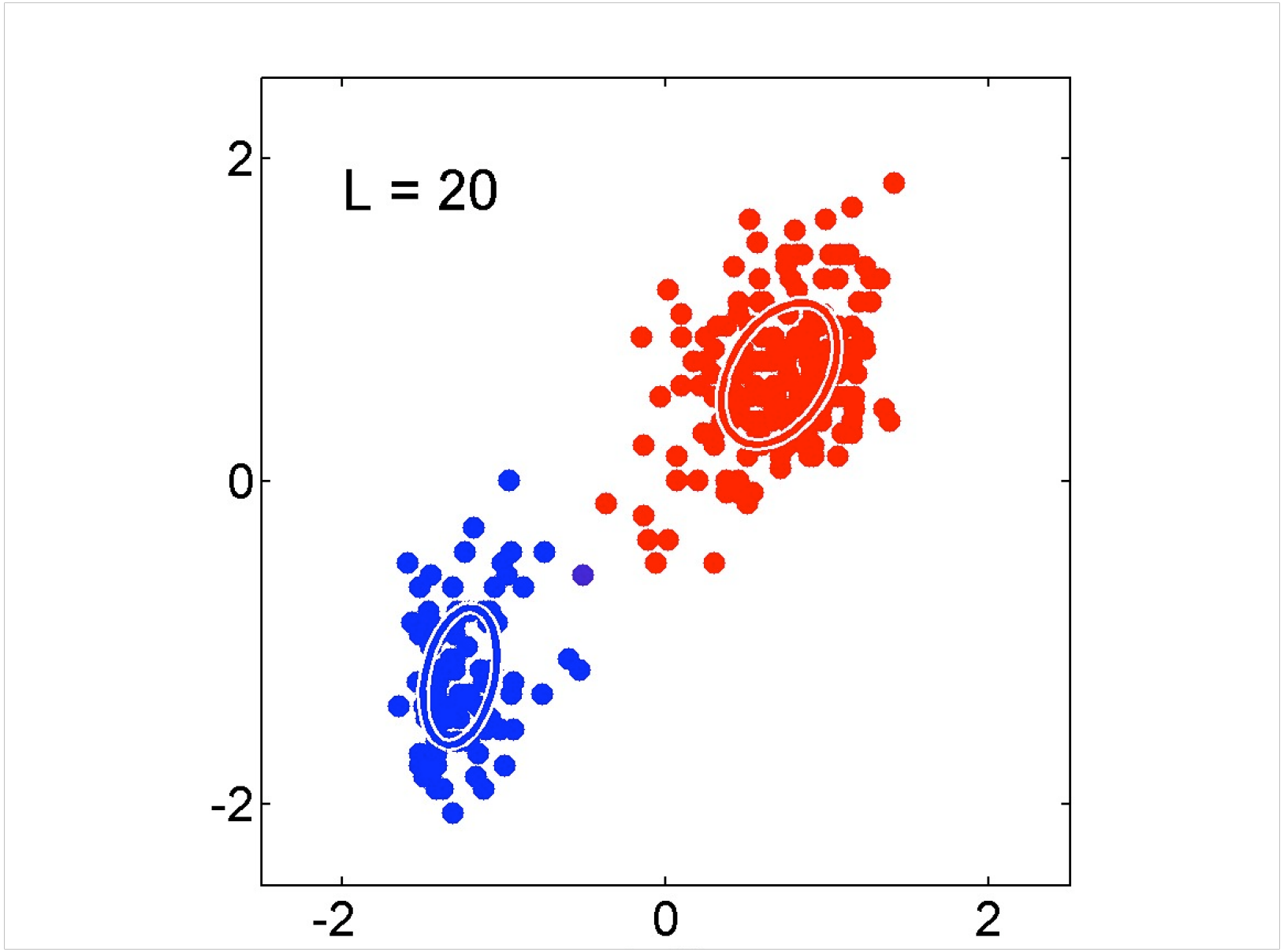

2
0
-2
-2
0
2


2
0
-2
-2
0
2

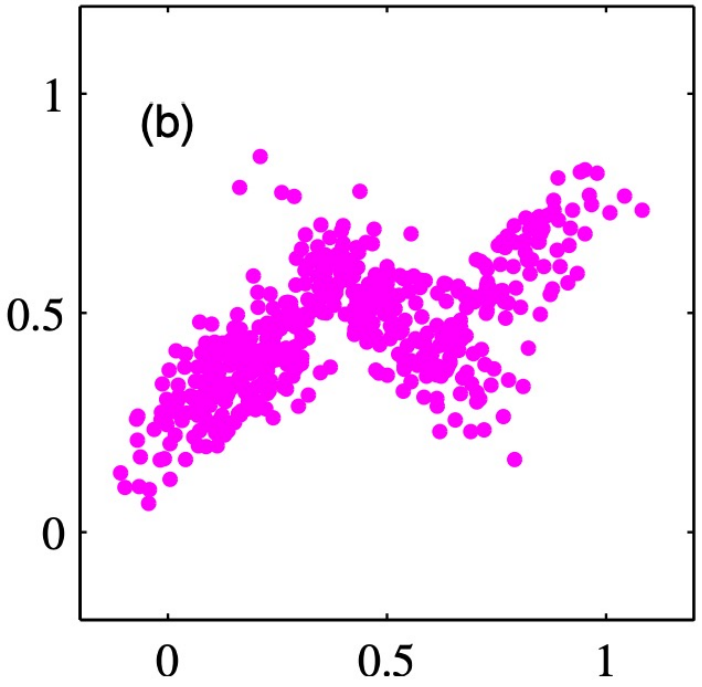

L = 1
2
0
-2
-2
0
2


L = 2
2
0
-2
-2
0
2

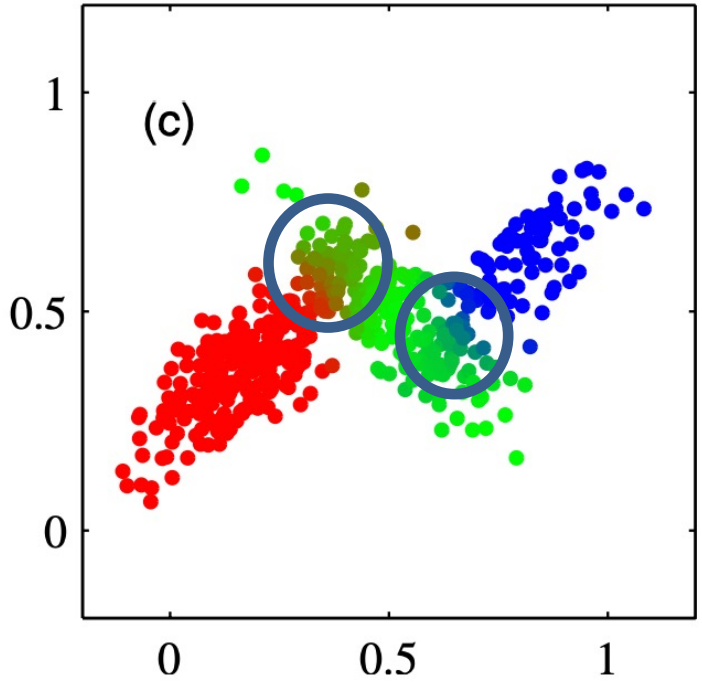

L = 5
2
0
-2
-2
0
2


L = 20
2
0
-2
-2
0
2

GMM with Soft EM Handles Overlapped Clusters



Unlabeled data



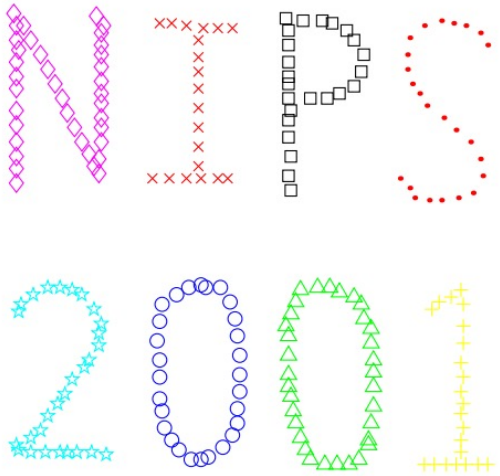
Clustering assignments via
GMM with soft EM

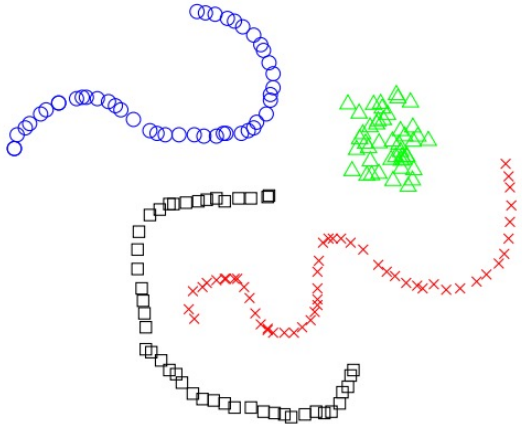
Summary

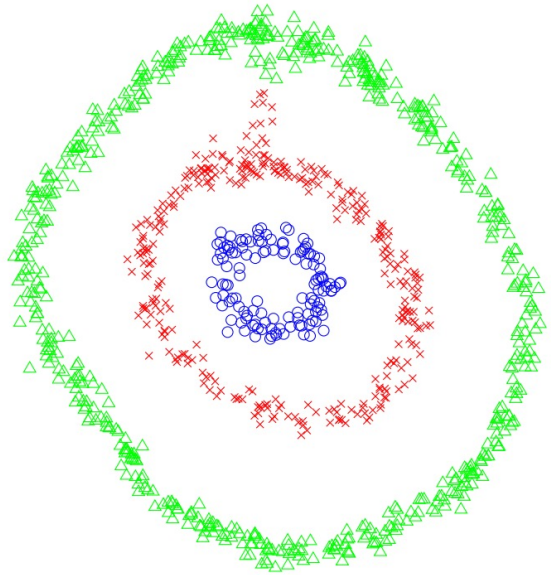
- Gaussian Mixture Model (GMM)
  - Sampling from a GMM
  - Fit a GMM: Log likelihood with latent variables
- Expectation Maximization (EM)
  - GMM with hard EM
    - Hard assignment: similar to K-means
    - Strong connection between GMM vs supervised GDA
    - K-means is a special case of hard EM
  - GMM with soft EM
    - Soft assignment: can address the overlapped clusters
  - Converge to local minimum

Limitations: K-means & GMM

- Mixture models and K-means focus on data **compactness**
- What about the data distributions like this?







Connectivity

Next Lecture:
Spectral Clustering

Acknowledgement

Some slides are from
Matt Gormley (CMU)
http://www.cs.cmu.edu/~mgormley/courses/10601/

Robert Collins (PSU)
http://www.cse.psu.edu/~rtc12/CSE586Spring2010/lectures/cse586gmmemPart2.pdf

Expectation Maximization: General Form

Given a joint distribution $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ over observed variables $\mathbf{X}$ and latent variables $\mathbf{Z}$, governed by parameters $\boldsymbol{\theta}$, the goal is to maximize the likelihood function $p(\mathbf{X}|\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$.

1. Choose an initial setting for the parameters $\boldsymbol{\theta}^{\text{old}}$.
2. **E step** Evaluate $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}})$.
3. **M step** Evaluate $\boldsymbol{\theta}^{\text{new}}$ given by

$$\boldsymbol{\theta}^{\text{new}} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}})$$

where

$$\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}}) \ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}).$$

4. Check for convergence of either the log likelihood or the parameter
   If the convergence criterion is not satisfied, then let

$$\boldsymbol{\theta}^{\text{old}} \leftarrow \boldsymbol{\theta}^{\text{new}}$$

   and return to step 2.