

# Expectation Maximization (EM) Algorithm

Machine Learning Course - CS-433

25 Nov 2025

Robert West

(Slide credits: Nicolas Flammarion)



# Gaussian Mixture Models

Data is modeled as a mixture of  $K$  Gaussian distributions

Instance space is  $\mathcal{X} = \underline{\mathbb{R}^D}$  and each sample  $\mathbf{x}$  is generated as follows:

1. Choose a random component  $k \in \{1, 2, \dots, K\}$ , i.e., let  $z$  be a multinomial r.v. with  $p(z = k) = \underline{\pi_k}$
2. Sample  $\mathbf{x}$ , given the value of  $z = k$ , from a Gaussian distribution:

$$p(\mathbf{x} | z = k) = \frac{1}{(2\pi)^{D/2} |\Sigma_k|^{1/2}} \exp \left( -\frac{1}{2} (\mathbf{x} - \underline{\mu_k})^T \underline{\Sigma_k^{-1}} (\mathbf{x} - \mu_k) \right)$$

⚠

# Gaussian Mixture Models

Then we can write the distribution of  $\mathbf{x}$  as:

$$\begin{aligned} p(\mathbf{x}) &= \sum_{k=1}^K p(z=k) p(\mathbf{x} | z=k) \\ &= \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \mu_k, \Sigma_k) \\ &= \sum_{k=1}^K \frac{\pi_k}{(2\pi)^{D/2} |\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_k)^T \Sigma_k^{-1} (\mathbf{x} - \mu_k)\right) \end{aligned}$$

$z$  is a latent (hidden) variable that is not directly observed

We introduce  $z$  because it helps us describe  $p(\mathbf{x})$  in a simple parametric form

# Gaussian Mixture Models

Then we can write the distribution of  $\mathbf{x}$  as:

$$\begin{aligned} p(\mathbf{x}) &= \sum_{k=1}^K p(z = k) p(\mathbf{x} | z = k) \\ &= \sum_{k=1}^K \pi_k \mathcal{N}(x | \mu_k, \Sigma_k) \\ &= \sum_{k=1}^K \frac{\pi_k}{(2\pi)^{D/2} |\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_k)^T \Sigma_k^{-1} (\mathbf{x} - \mu_k)\right) \end{aligned}$$

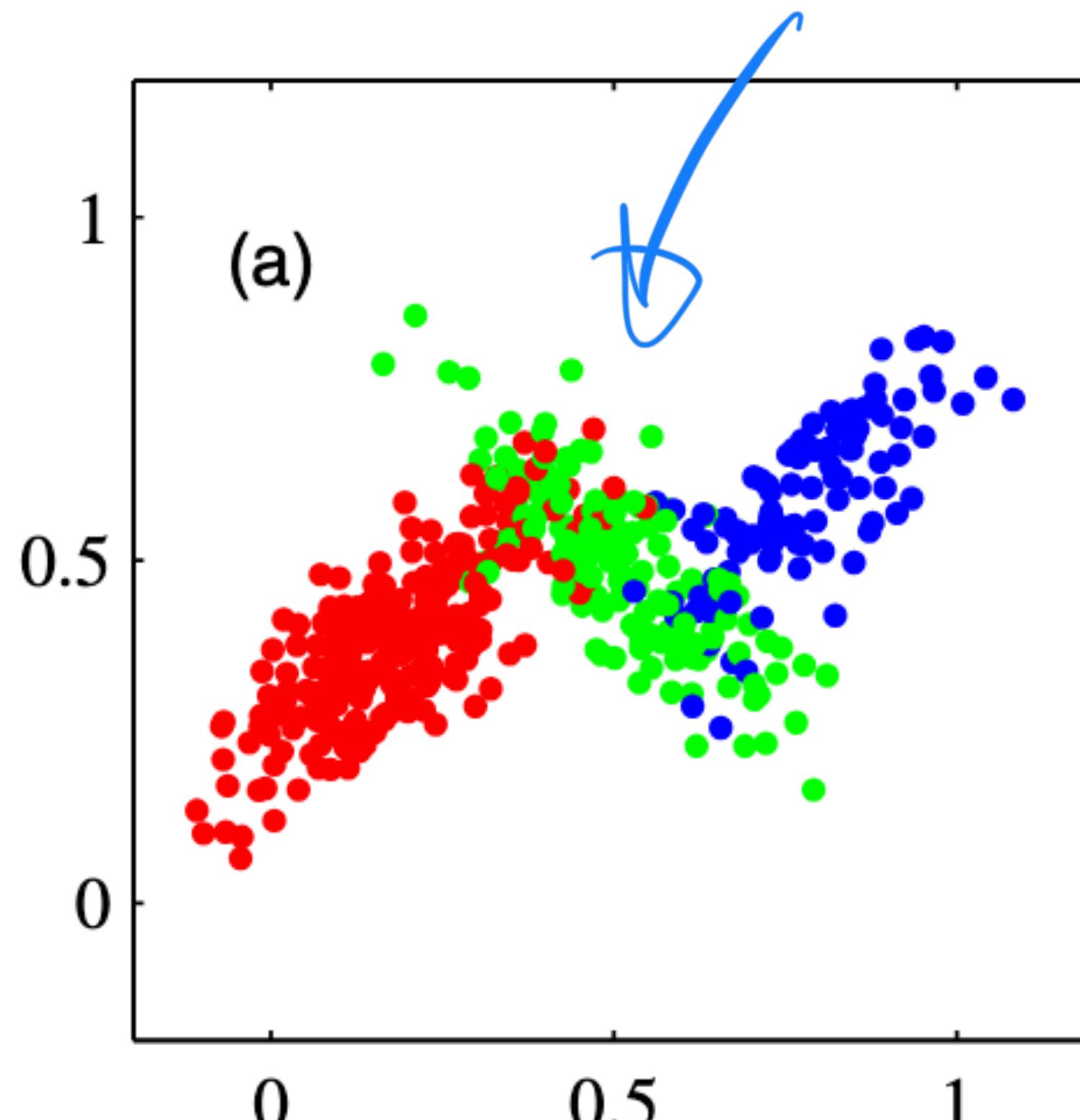
mixing proportion

mixture components

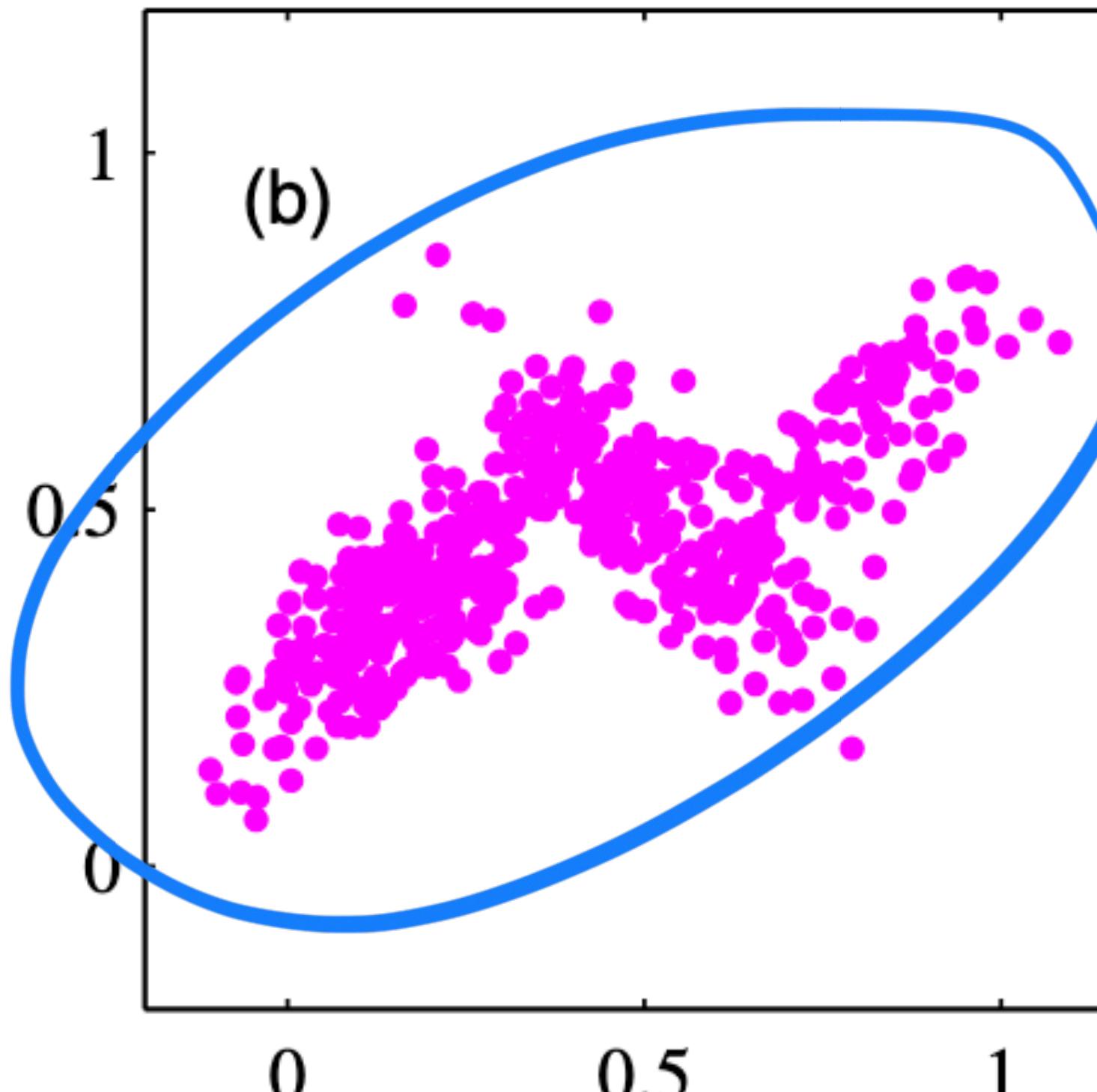
$z$  is a latent (hidden) variable that is not directly observed

We introduce  $z$  because it helps us describe  $p(\mathbf{x})$  in a simple parametric form

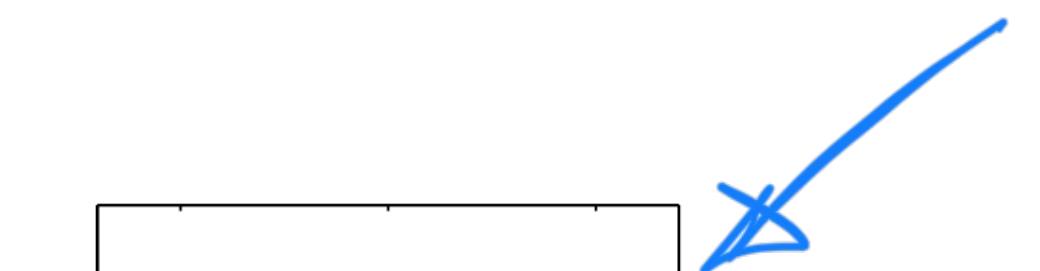
# Visualizing the Joint and Marginal Distributions of GMM



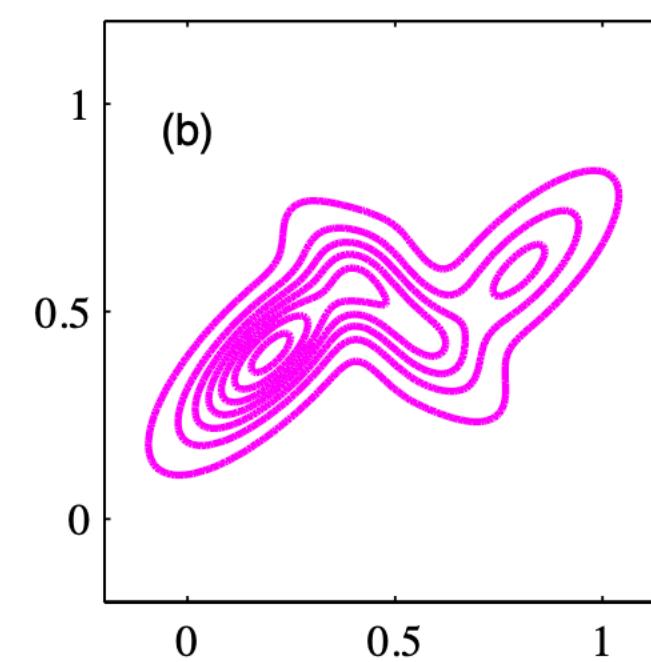
Samples from the joint distribution  $p(\mathbf{x}, z)$ , with  $z$  represented by colors



Samples from the marginal distribution  $p(\mathbf{x})$



Joint distribution  $p(\mathbf{x}, z)$ , with  $z$  represented by colors



Marginal distributions  $p(\mathbf{x})$

# MLE with Latent Variables

latent  
variables  
model  
param  
unobserved

Given an i.i.d. sample  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$  and a latent variable model  $p(\mathbf{x}, z | \theta)$ , we want to maximize the log-likelihood of  $\mathcal{D}$ , i.e.  $\hat{\theta}_{MLE} = \arg \max_{\theta} \log p(\mathcal{D} | \theta)$ :

$$\begin{aligned}\log p(\mathcal{D} | \theta) &= \log \prod_{n=1}^N p(\mathbf{x}_n | \theta) \\ &= \sum_{n=1}^N \log p(\mathbf{x}_n | \theta) \quad \text{latent variables } (z_1, \dots, z_N) \\ &= \sum_{n=1}^N \log \left( \sum_{z_n=1}^K p(\mathbf{x}_n, z_n | \theta) \right) \\ &= \sum_{n=1}^N \log \left( \sum_{z_n=1}^K p(z_n | \theta) p(\mathbf{x}_n | z_n, \theta) \right)\end{aligned}$$

# MLE with Latent Variables

Given an i.i.d. sample  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$  and a latent variable model  $p(\mathbf{x}, z | \theta)$ , we want to maximize the log-likelihood of  $\mathcal{D}$

$$\begin{aligned}\log p(\mathcal{D} | \theta) &= \log \prod_{n=1}^N p(\mathbf{x}_n | \theta) \\ &= \sum_{n=1}^N \log p(\mathbf{x}_n | \theta) \\ &= \sum_{n=1}^N \log \left( \sum_{z_n=1}^K p(\mathbf{x}_n, z_n | \theta) \right) \\ &= \sum_{n=1}^N \log \left( \sum_{z_n=1}^K p(z_n | \theta) p(\mathbf{x}_n | z_n, \theta) \right)\end{aligned}$$

In many cases, the summation inside the log makes the optimization problem computationally hard. The Expectation-Maximization (EM) is designed for cases in which, had we known the values of the latent variables  $z$ , the MLE would have been computationally tractable.

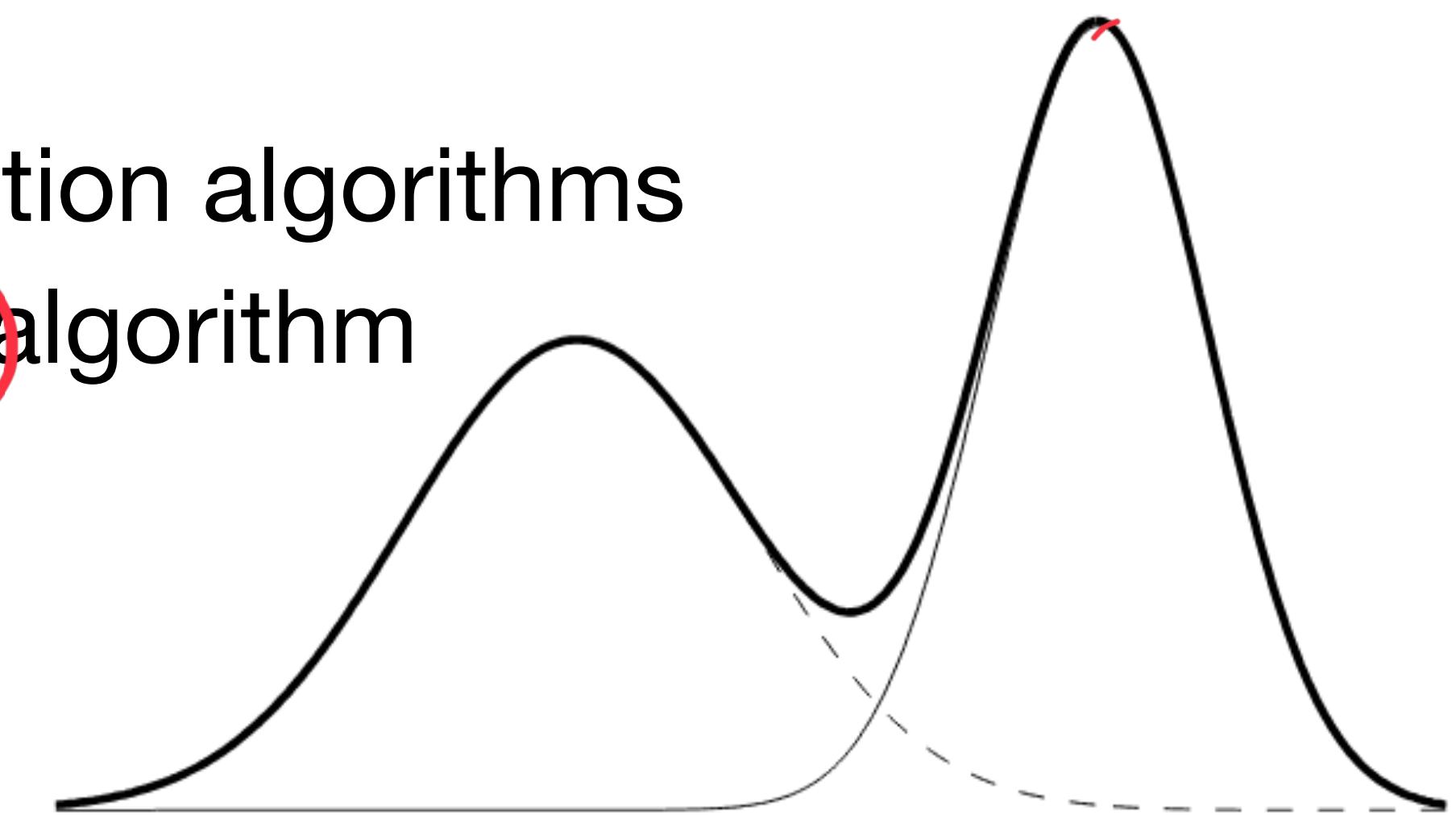
unobserved



# MLE Objective for Gaussian Mixtures

$$\max_{\theta} \sum_{n=1}^N \log \left( \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \right)$$

- This is a **non-convex** objective
- There is **no unique optimum** (permutation of K clusters)
- The objective is **unbounded**
- Can be solved directly using black-box optimization algorithms
- Main focus: How to solve this problem with **EM** algorithm



Average of two probability distributions of two Gaussian  
for which it is natural to introduce a mixture model

# MLE with access to Labels for GMM

Assume observing the *complete data*  $\mathcal{D}_c = \{(\mathbf{x}_n, z_n)\}_{n \in [N]}$ , the *complete log likelihood* is:

$$\pi_k = \frac{e^{\alpha_k}}{\sum_{k'} e^{\alpha_{k'}}}$$

$$\mathcal{L}_c(\theta | \mathcal{D}_c) = \sum_{n=1}^N \log(\pi_{z_n} \mathcal{N}(\mathbf{x}_n | \mu_{z_n}, \Sigma_{z_n}))$$

$\overbrace{p(\mathcal{D}_c | \theta)}$

→ Convex!

We could easily compute MLE in closed form:

$$\hat{\pi}_k = \frac{1}{N} \sum_n \mathbf{1}\{z_n = k\}$$

$$\hat{\mu}_k = \frac{\sum_n \mathbf{1}\{z_n = k\} \mathbf{x}_n}{\sum_n \mathbf{1}\{z_n = k\}}$$

$$\hat{\Sigma}_k = \frac{1}{\sum_n \mathbf{1}\{z_n = k\}} \sum_n \mathbf{1}\{z_n = k\} (\mathbf{x}_n - \hat{\mu}_k)(\mathbf{x}_n - \hat{\mu}_k)^T$$



→ If we had access to the **labels**, the problem simplifies

# Hard EM Algorithm

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

Given dataset  $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ , initialize the parameters  $\theta^{(0)} = \{\pi_k, \mu_k, \Sigma_k\}_{k \in [K]}$

For  $t = 1, 2, \dots$

**E-step:** Predict the most likely class for each data point

$$\begin{aligned} z_n^{(t)} &= \arg \max_z p(z | \mathbf{x}_n, \theta^{(t-1)}) = \arg \max_z p(z | \theta^{(t-1)}) p(\mathbf{x}_n | z, \theta^{(t-1)}) \\ &= \arg \max_z \pi_z \mathcal{N}(\mathbf{x}_n | \mu_z, \Sigma_z) \end{aligned}$$

Bayes rule

Now we have **complete labeled data!** Let  $\mathcal{D}_c^{(t)} = \{(\mathbf{x}_n, z_n^{(t)})\}_{n \in [N]}$

**M-step:** Compute MLE of  $\theta$  using  $\mathcal{D}_c^{(t)}$

$$\theta^{(t)} = \arg \max_{\theta} p(\mathcal{D}_c^{(t)} | \theta) = \arg \max_{\theta} \sum_{n=1}^N \log(\pi_{z_n^{(t)}} \mathcal{N}(\mathbf{x}_n | \mu_{z_n^{(t)}}, \Sigma_{z_n^{(t)}}))$$

# Hard EM & K-Means

**K-Means is a special case of Hard-EM** where optimization is over the means  $\mu_k$  with fixed uniform weights ( $\pi_k = 1/K \forall k$ ) and fixed identical spherical covariances ( $\Sigma_k = \sigma^2 I \forall k$ ).

E-step:

$$z_n^{(t)} = \arg \max_k \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) = \arg \max_k \frac{1}{K} \frac{1}{\sqrt{(2\pi\sigma^2)^d}} \exp(-\|\mathbf{x}_n - \mu_k\|^2 / 2\sigma^2)$$
$$= \arg \min_k \|\mathbf{x}_n - \mu_k\|^2$$

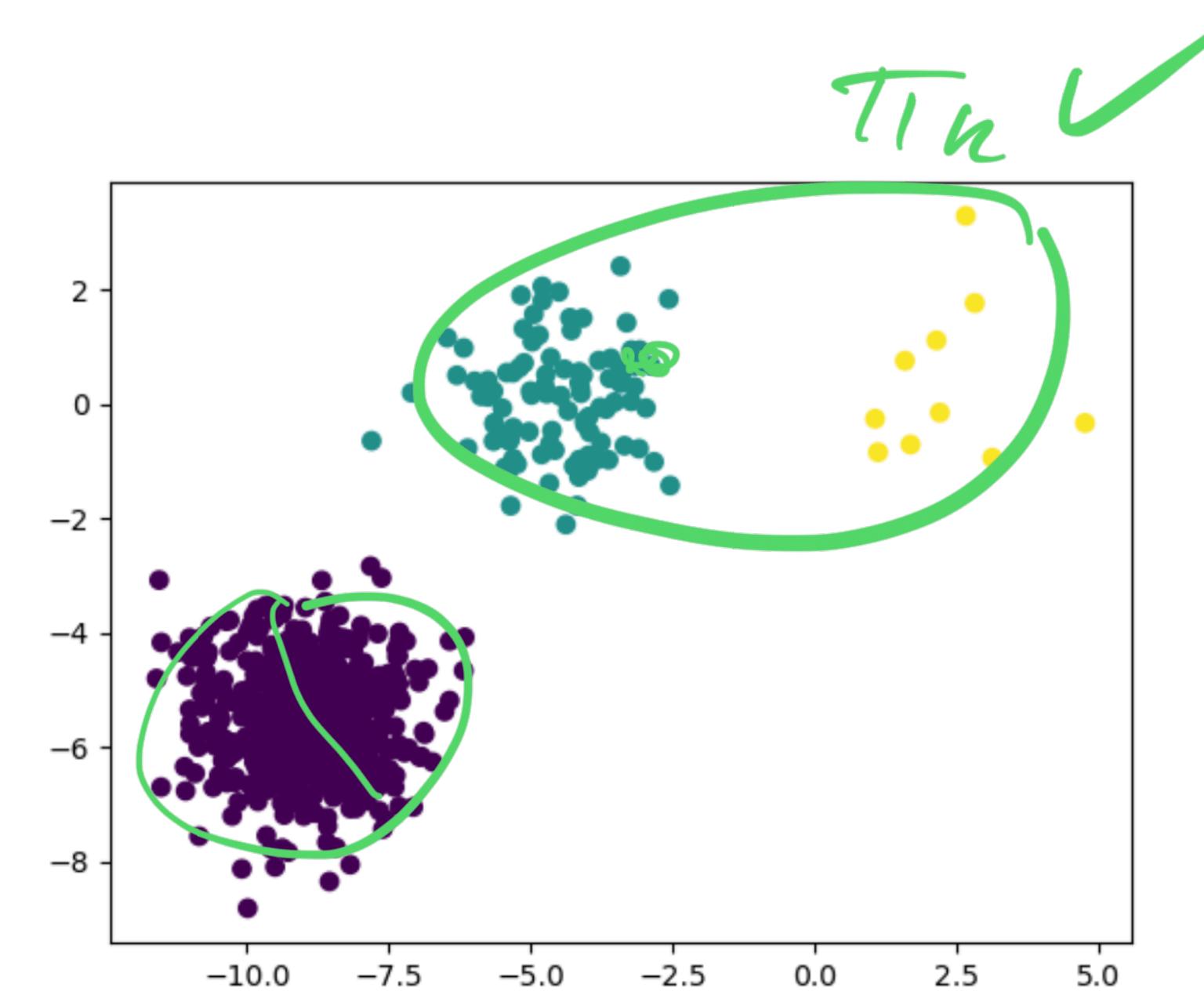
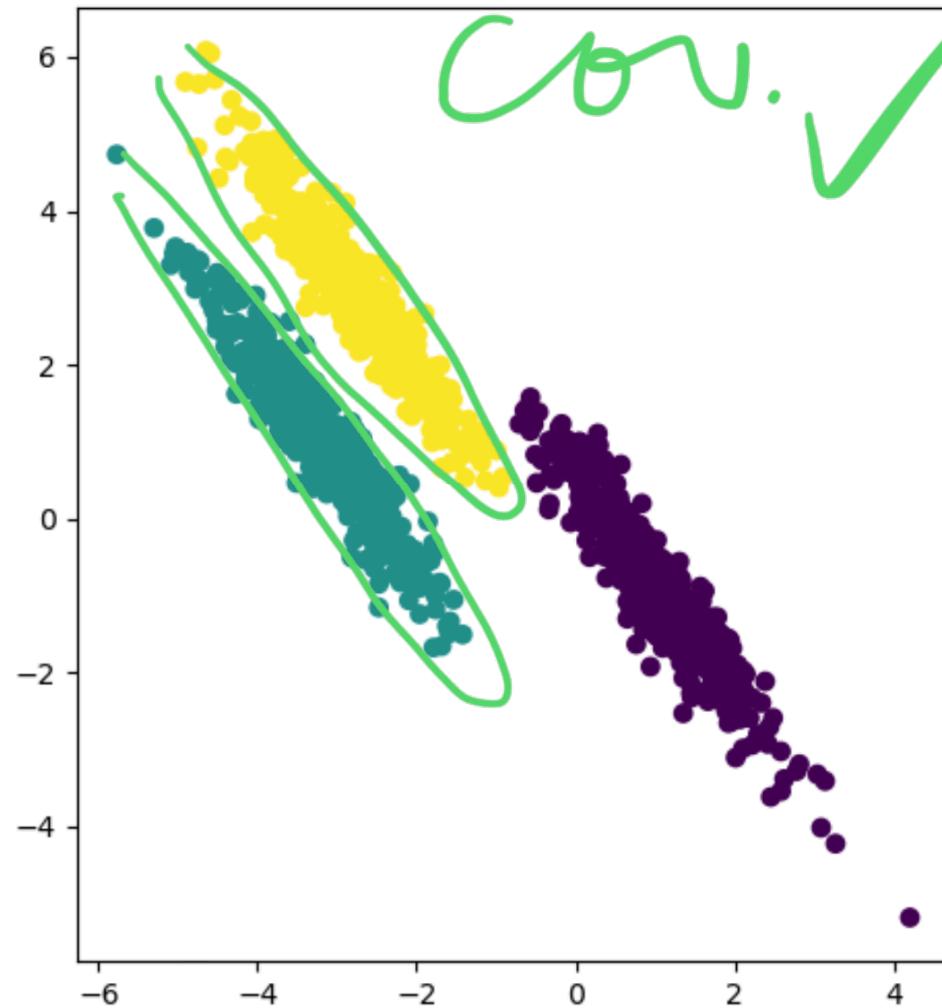
M-step:

$$\theta^{(t)} = \arg \max_{\theta} \sum_{n=1}^N \log(\pi_{z_n^{(t)}} \mathcal{N}(\mathbf{x}_n | \mu_{z_n^{(t)}}, \Sigma_{z_n^{(t)}})) = \arg \min_{\theta} \sum_{n=1}^N \|\mathbf{x}_n - \mu_{z_n^{(t)}}\|^2$$
$$\Rightarrow \mu_k^{(t)} = \frac{1}{\sum_n \mathbf{1}\{z_n^{(t)} = k\}} \sum_n \mathbf{1}\{z_n^{(t)} = k\} \mathbf{x}_n$$

# Hard EM vs K-Means

Hard-EM handles clusters of **different shapes and sizes**, unlike K-means

It allows flexible choices of  $\Sigma_k$  and  $\pi_k$



However, similar to K-means, Hard-EM assigns fixed labels to points, even when the model is uncertain

# Posterior Probabilities for Latent Variables in GMM

For each data point, we compute the posterior over cluster memberships given the parameters  $\theta = \{\pi_k, \mu_k, \Sigma_k\}_{k \in [K]}$

This models distributions over latent variables  $z$  rather than assigning a single (hard) label

$$q_k(\mathbf{x}_n) = p(z_n = k | \mathbf{x}_n, \theta)$$

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

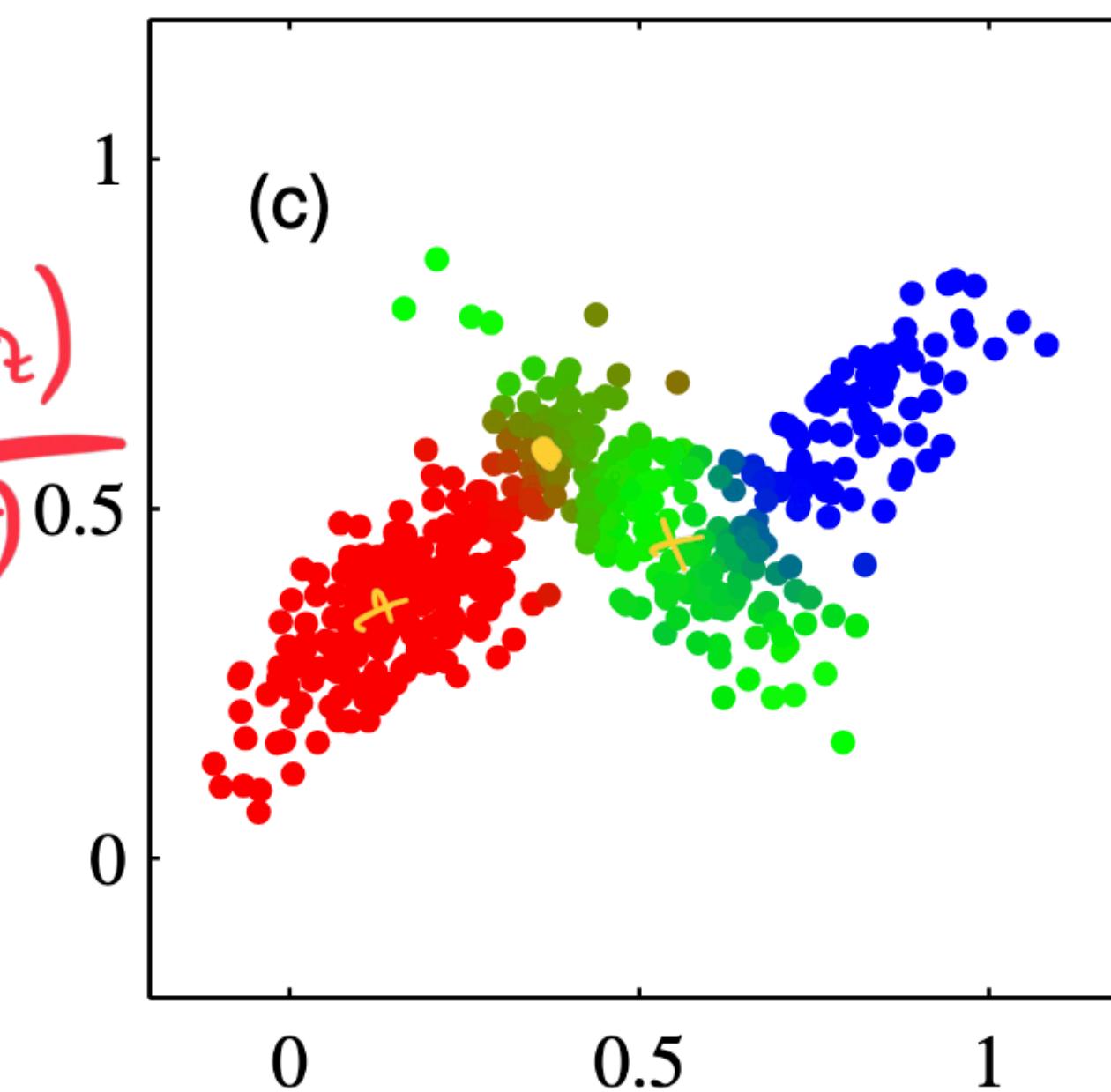
*Bayes' rule*

*Marginalize*

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

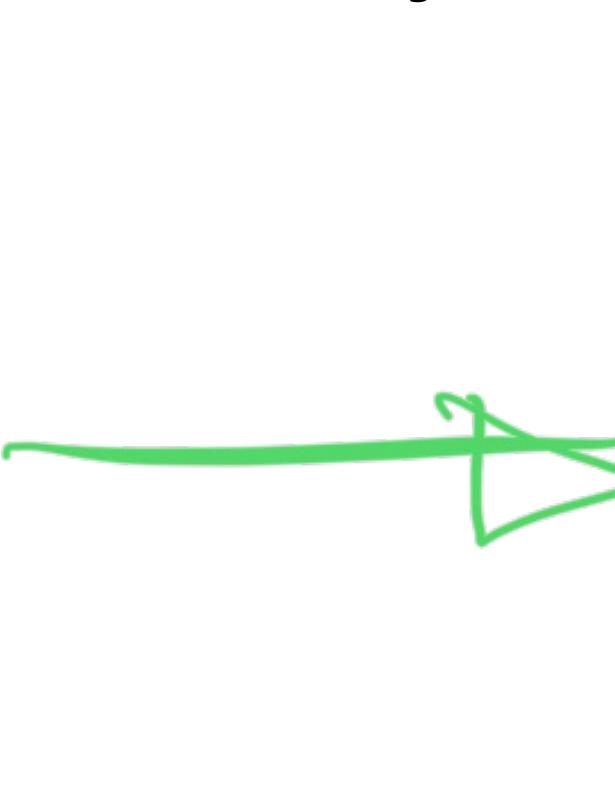
$$p(x) = \sum_k p(x|z=k)p(z=k)$$

Soft assignments capture uncertainty in cluster memberships (unlike K-means)



# MLE for GMM: Coupled Parameter Updates

The MLE estimates  $(\mu^\star, \Sigma^\star, \pi^\star) = \arg \max_{(\mu, \Sigma, \pi)} \sum_{n=1}^N \log \left( \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \right)$  satisfy:


$$\pi_k^\star = \frac{1}{N} \sum_{n=1}^N q_k(\mathbf{x}_n)$$
$$\mu_k^\star = \frac{\sum_{n=1}^N q_k(\mathbf{x}_n) \mathbf{x}_n}{\sum_{n=1}^N q_k(\mathbf{x}_n)}$$
$$\Sigma_k^\star = \frac{\sum_{n=1}^N q_k(\mathbf{x}_n) (\mathbf{x}_n - \mu_k^\star) (\mathbf{x}_n - \mu_k^\star)^\top}{\sum_{n=1}^N q_k(\mathbf{x}_n)}$$

$\overbrace{1} \Rightarrow q$

**Chicken-and-egg problems: These equations are coupled!**  
**Solving them jointly is difficult.**

# MLE for GMM: Coupled Parameter Updates

The MLE estimates  $(\mu^\star, \Sigma^\star, \pi^\star) = \arg \max$  satisfy:

$$\pi_k^\star = \frac{1}{N} \sum_{n=1}^N q_k(\mathbf{x}_n)$$

$$\Sigma_k^\star = \frac{\sum_{n=1}^N q_k(\mathbf{x}_n) (\mathbf{x}_n - \mu_k)(\mathbf{x}_n - \mu_k)^T}{\sum_{n=1}^N q_k(\mathbf{x}_n)}$$

**Chicken-and-egg problems:** These equations are coupled.  
Solving them jointly is difficult.

Proof:

$$\begin{aligned}\frac{\partial l}{\partial \mu_k} &= \frac{\partial}{\partial \mu_k} \sum_{n=1}^N \log \left( \sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j) \right) \\ &= \sum_{n=1}^N \frac{\pi_k}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)} \frac{\partial}{\partial \mu_k} \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \\ &= \sum_{n=1}^N \frac{q_k(\mathbf{x}_n)}{\mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)} \frac{\partial}{\partial \mu_k} \mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k) \\ &= \sum_{n=1}^N q_k(\mathbf{x}_n) \Sigma_k^{-1} (\mathbf{x}_n - \mu_k)\end{aligned}$$

Setting to zero yields  $\mu_k^\star = \frac{\sum_{n=1}^N q_k(\mathbf{x}_n) \mathbf{x}_n}{\sum_{n=1}^N q_k(\mathbf{x}_n)}$

# (Soft) EM Algorithm for GMMs

Initialize the parameters  $\theta^{(0)} = \{\pi_k, \mu_k, \Sigma_k\}_{k \in [K]}$ . For  $t = 1, 2, \dots$ :

**E-step:** Compute posterior cluster assignments

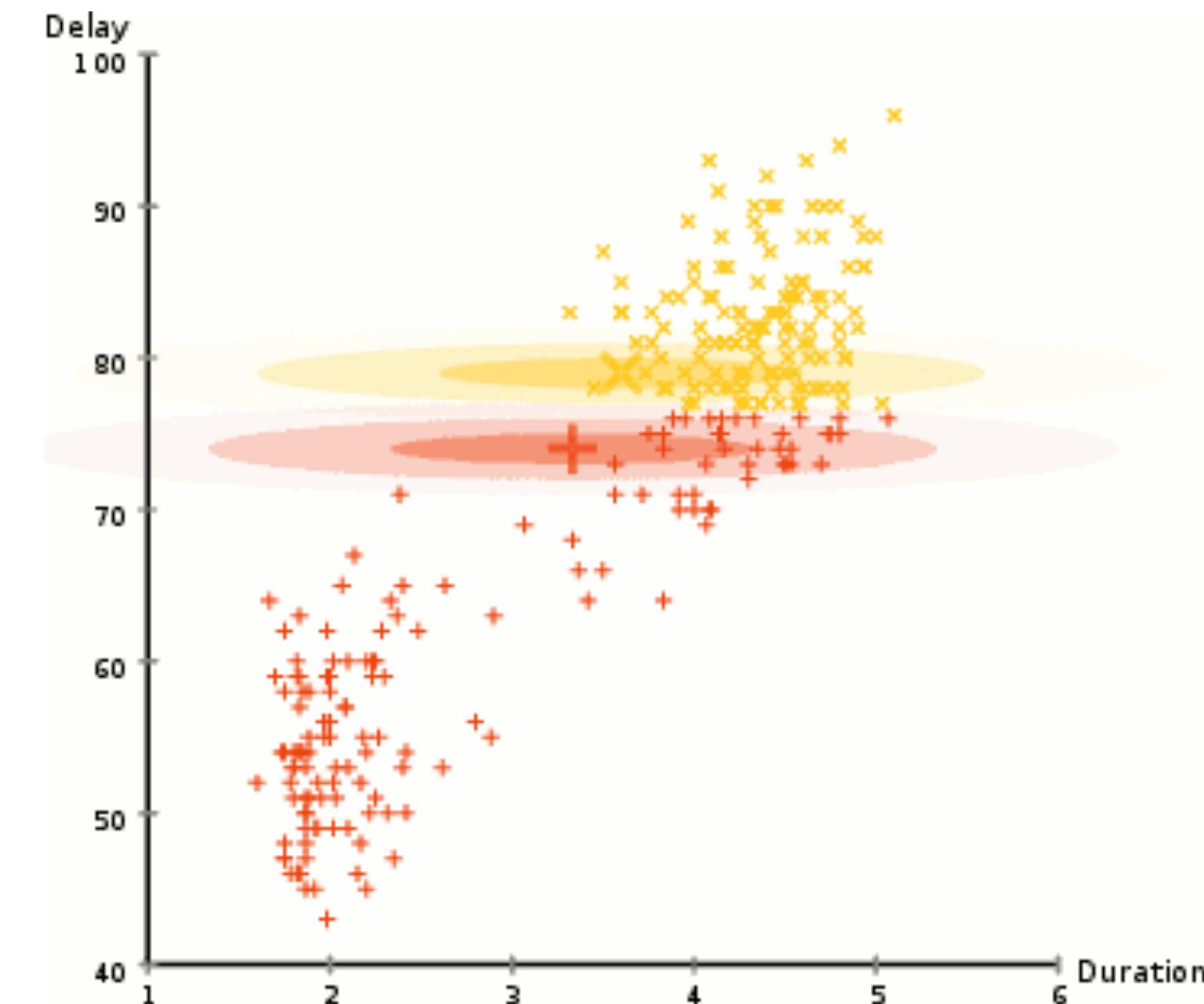
$$q_k^{(t)}(\mathbf{x}_n) = p(z_n = k | \mathbf{x}_n, \theta^{(t-1)}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n; \mu_k^{(t-1)}, \Sigma_k^{(t-1)})}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n; \mu_j^{(t-1)}, \Sigma_j^{(t-1)})}$$

**M-step:** Compute MLE for  $\theta$  using the calculated  $q_k^{(t)}(\mathbf{x}_n)$

$$\pi_k^{(t)} = \frac{1}{N} \sum_n q_k^{(t)}(\mathbf{x}_n), \quad \mu_k^{(t)} = \frac{\sum_{n=1}^N q_k^{(t)}(\mathbf{x}_n) \mathbf{x}_n}{\sum_{n=1}^N q_k^{(t)}(\mathbf{x}_n)}, \quad \Sigma_k^{(t)} = \frac{\sum_{n=1}^N q_k^{(t)}(\mathbf{x}_n) (\mathbf{x}_n - \mu_k^{(t)}) (\mathbf{x}_n - \mu_k^{(t)})^\top}{\sum_{n=1}^N q_k^{(t)}(\mathbf{x}_n)}$$

Iteration cost:  $O(NKD^3)$

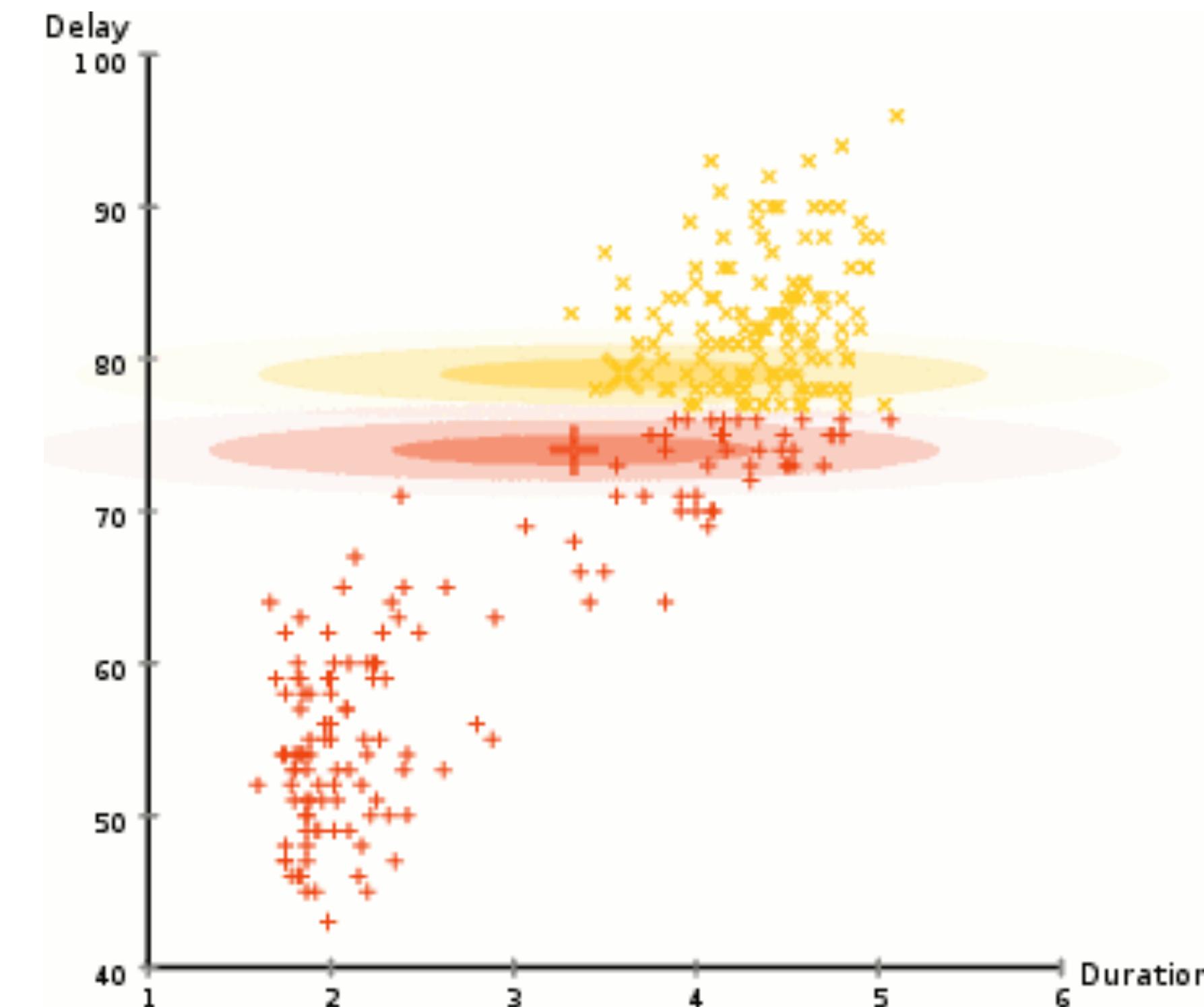
# (Soft) EM Algorithm



EM Clustering of Old Faithful data

(Video)

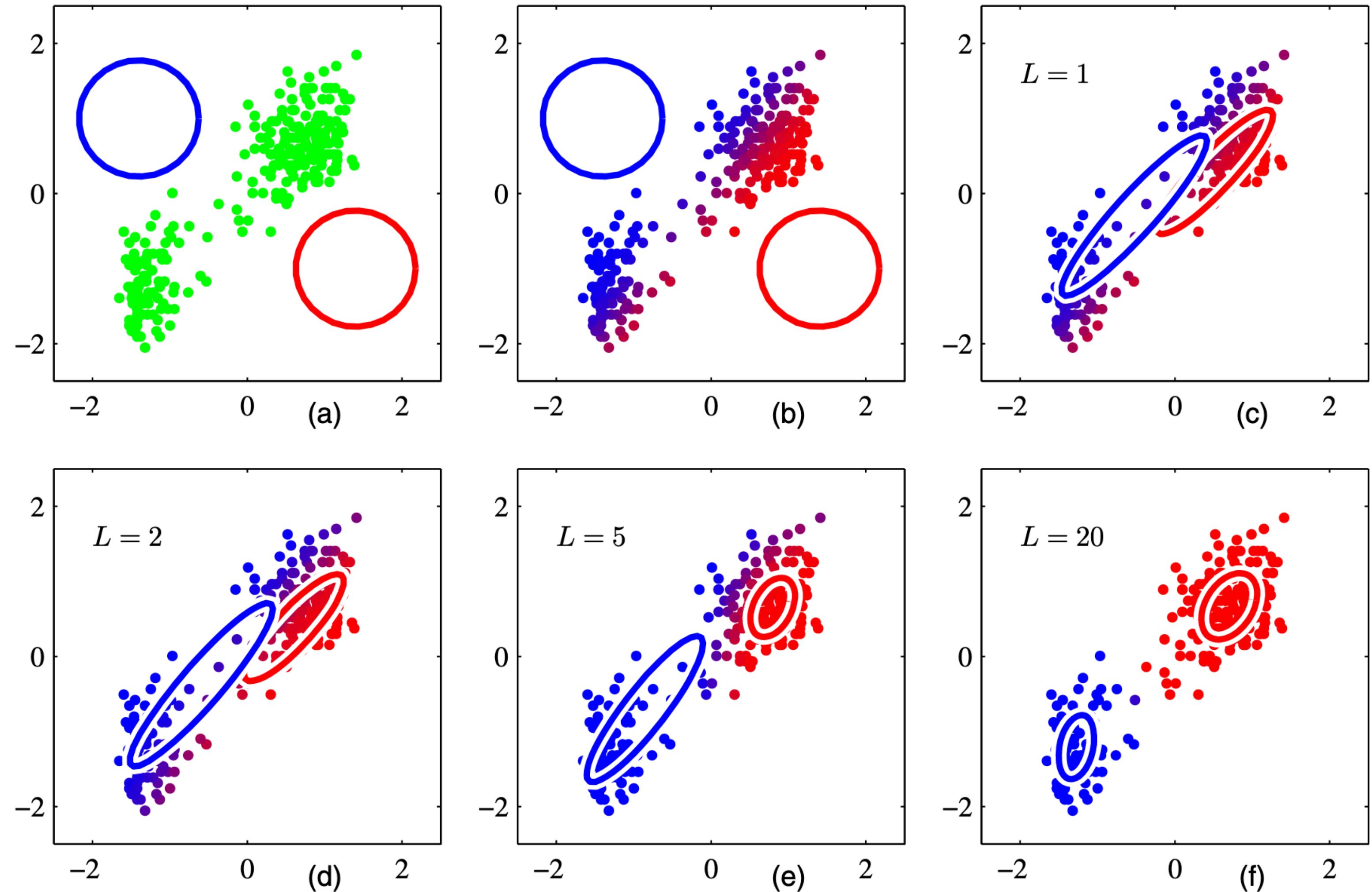
# (Soft) EM Algorithm



EM Clustering of Old Faithful data

(Video)

# (Soft) EM Algorithm



# Comparison of EM and K-Means

EM typically requires more iterations to converge compared to K-means

Each iteration is computationally more expensive ( $O(NKD^3)$  vs  $O(NKD)$ )

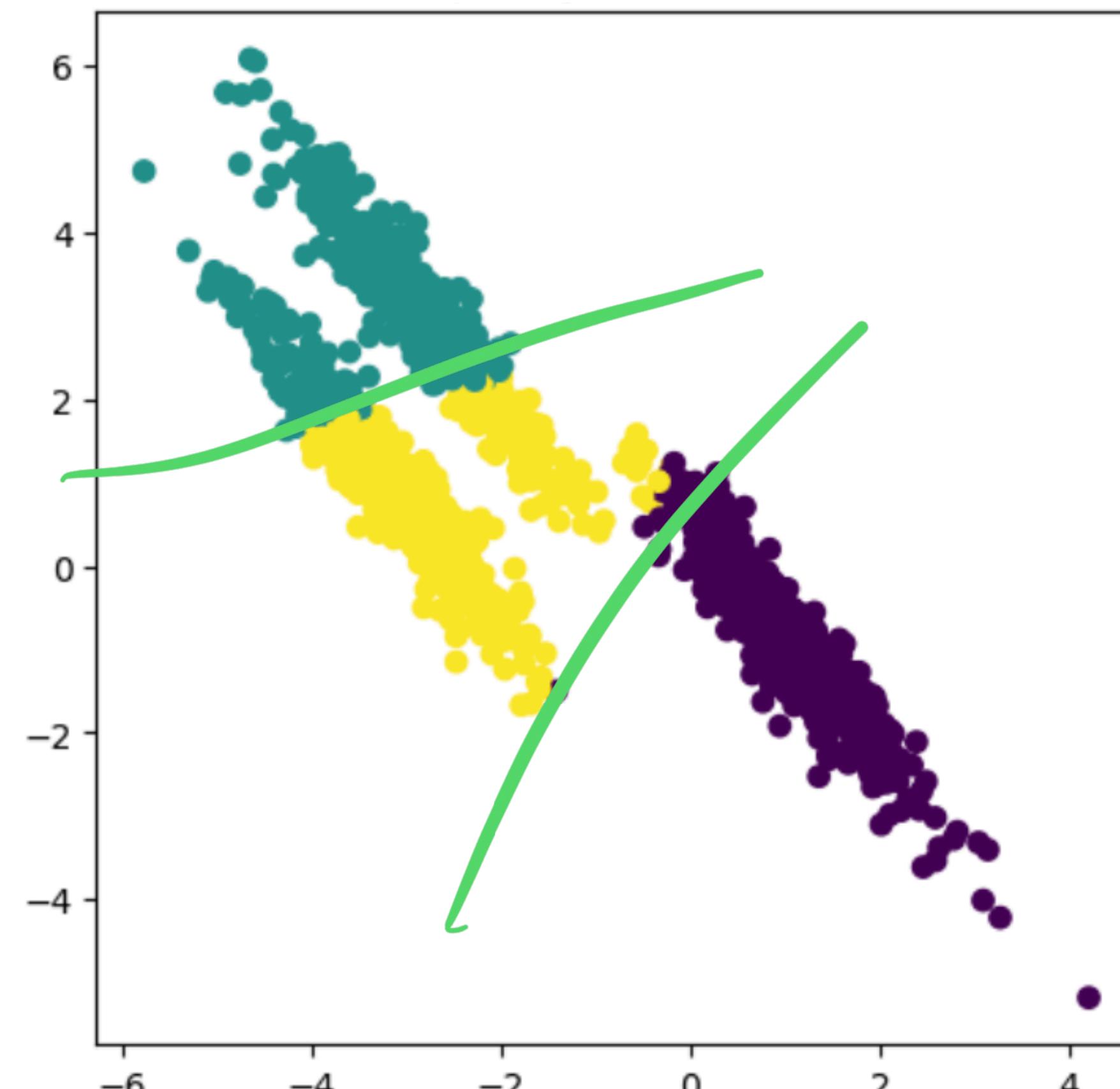
K-means is often used to initialize EM for faster convergence

EM outperforms K-means on challenging cases:

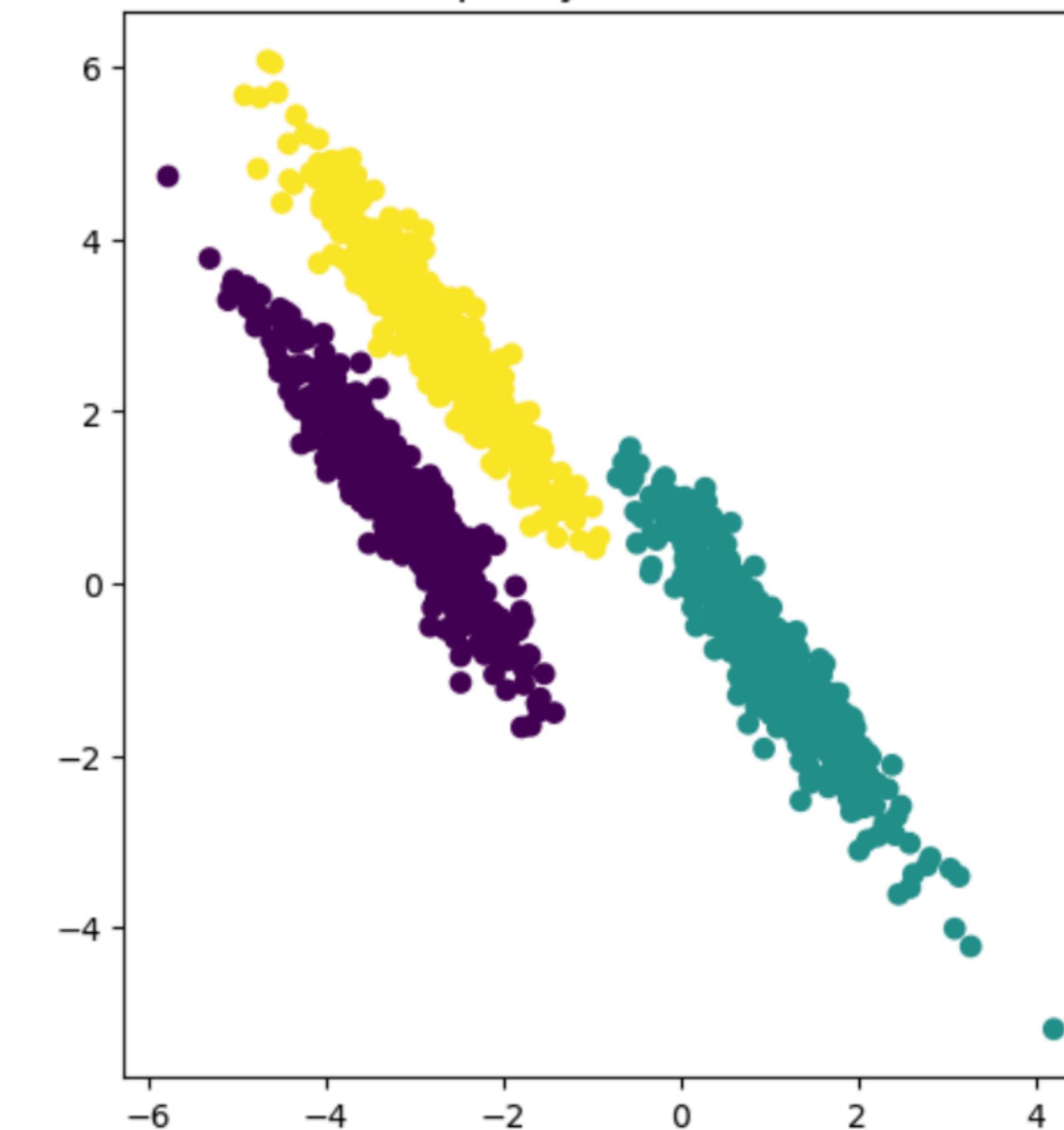
- Anisotropic clusters
- Unequal variances of clusters
- Unevenly sized clusters

→ EM is more flexible than K-means but is slower and more computationally expensive, making it better suited for complex clustering tasks

# EM vs K-Means: Anisotropic Clusters

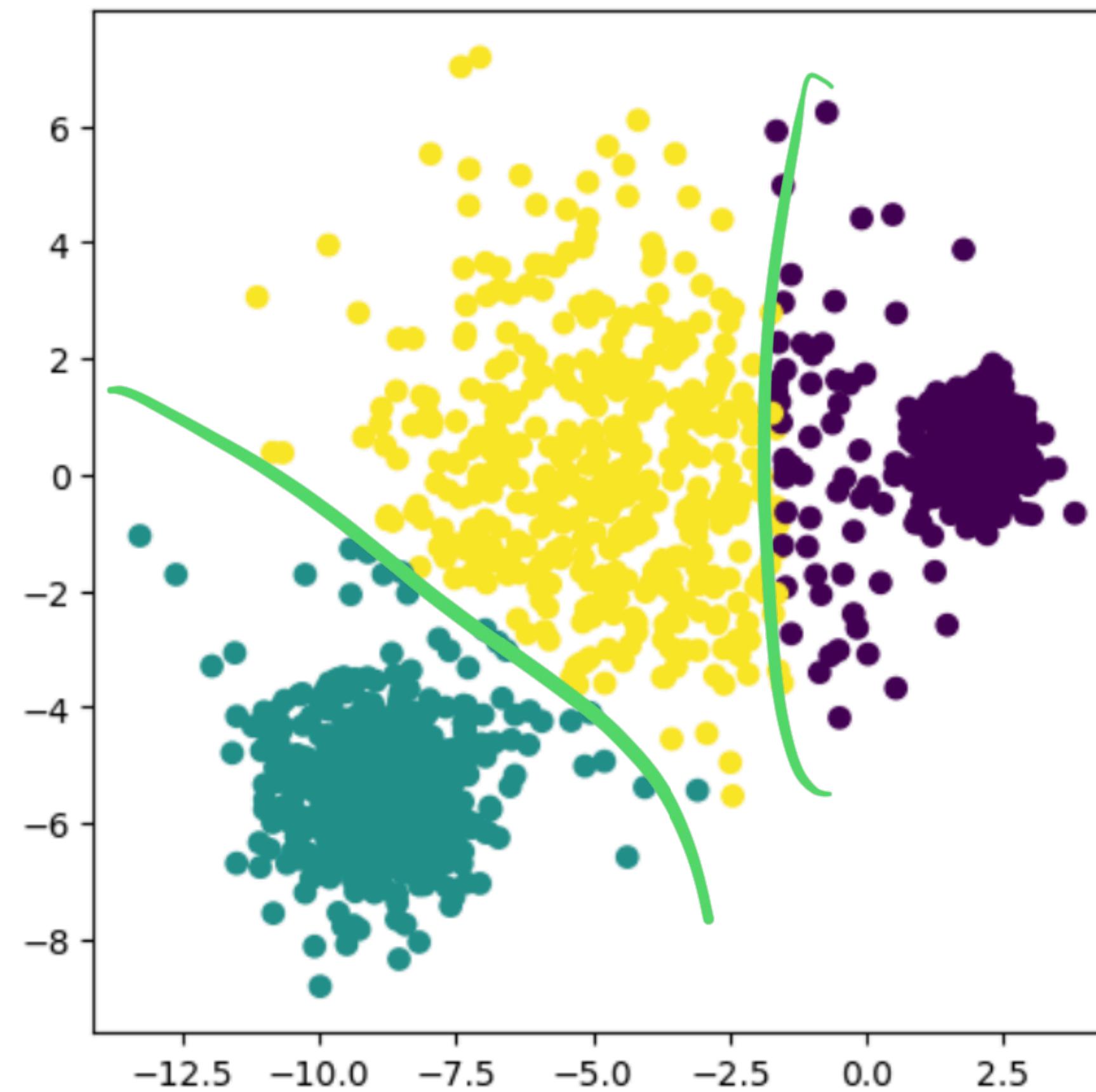


K-means

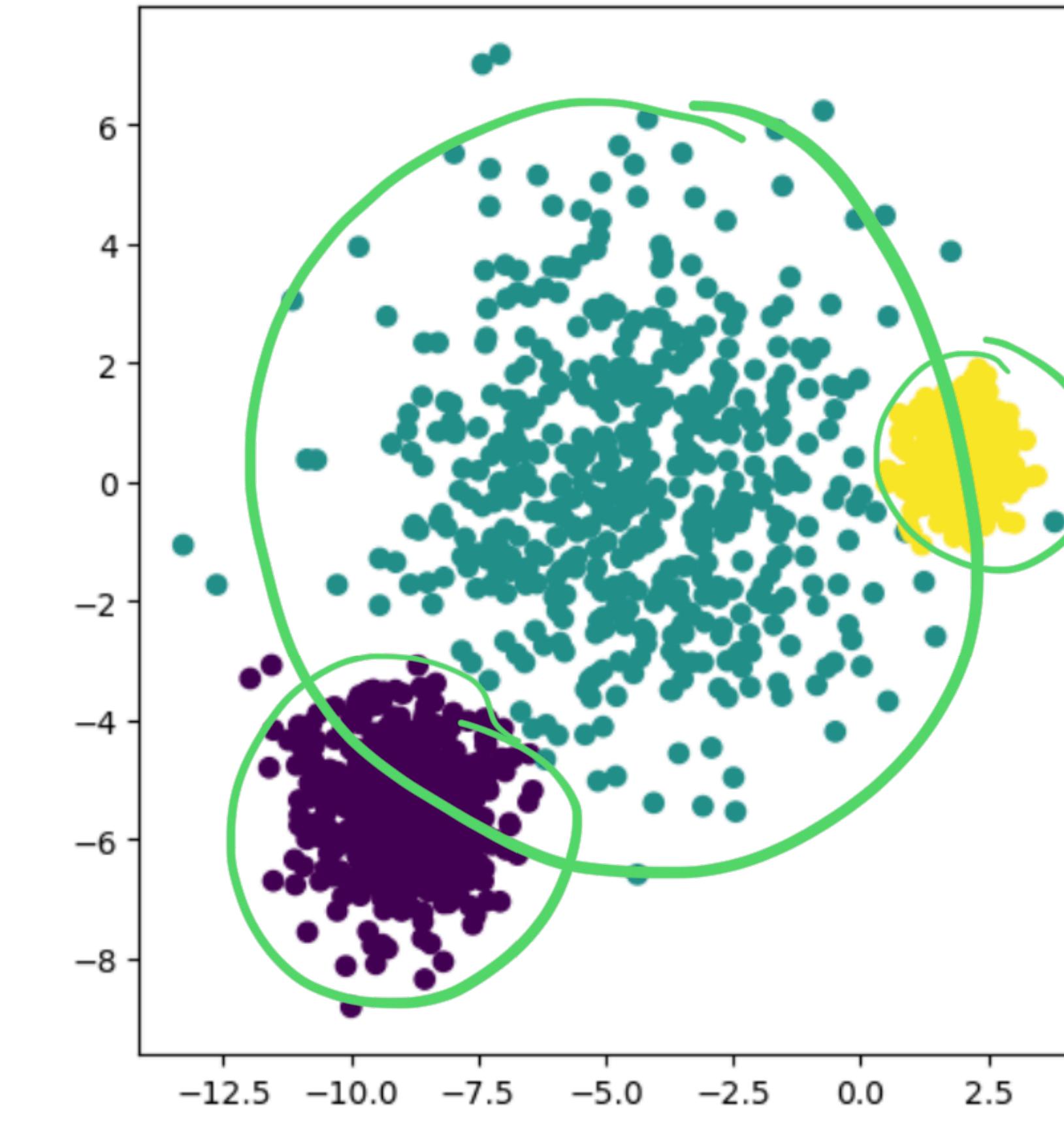


EM

# EM vs K-Means: Unequal Variances

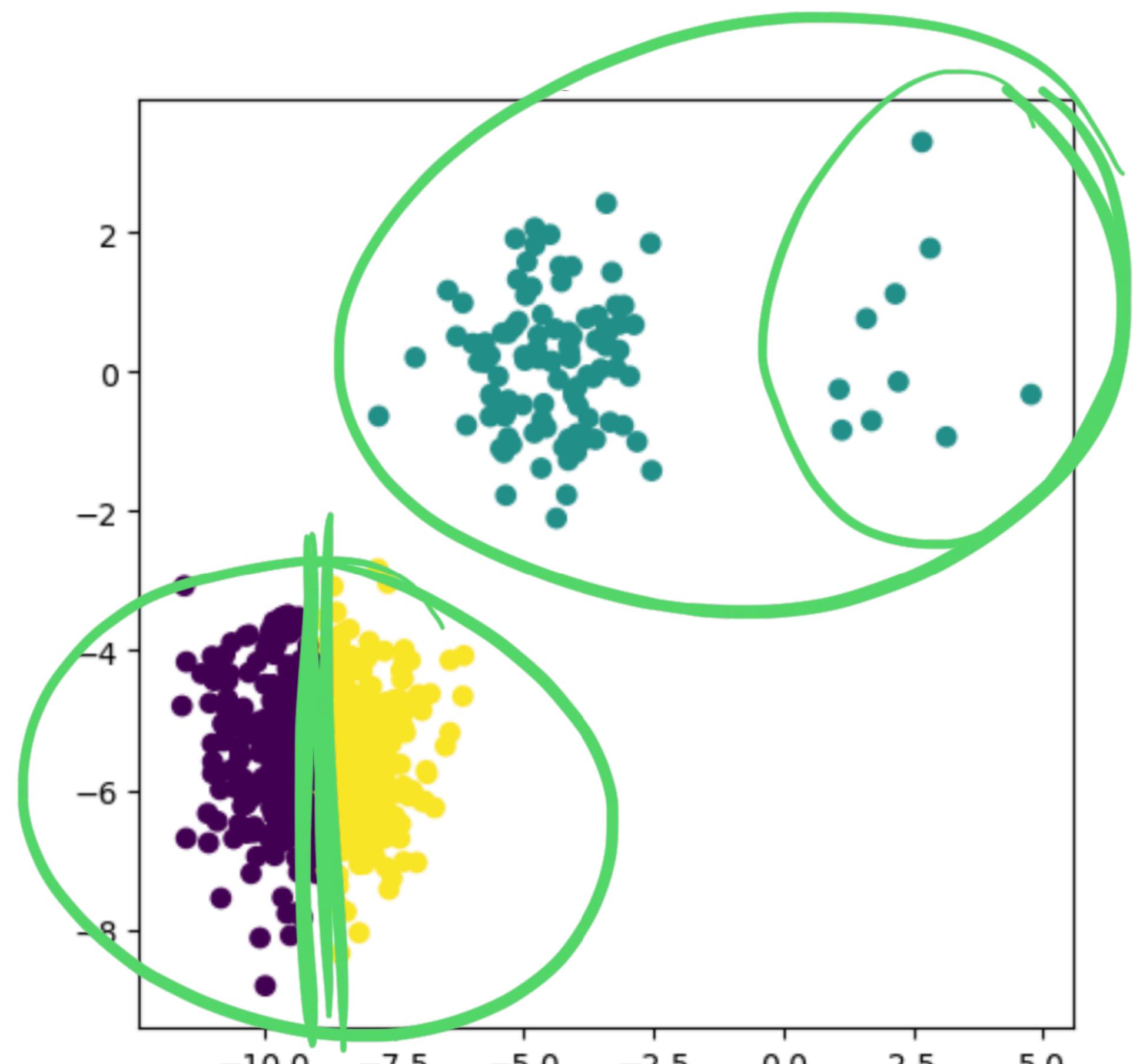


K-means

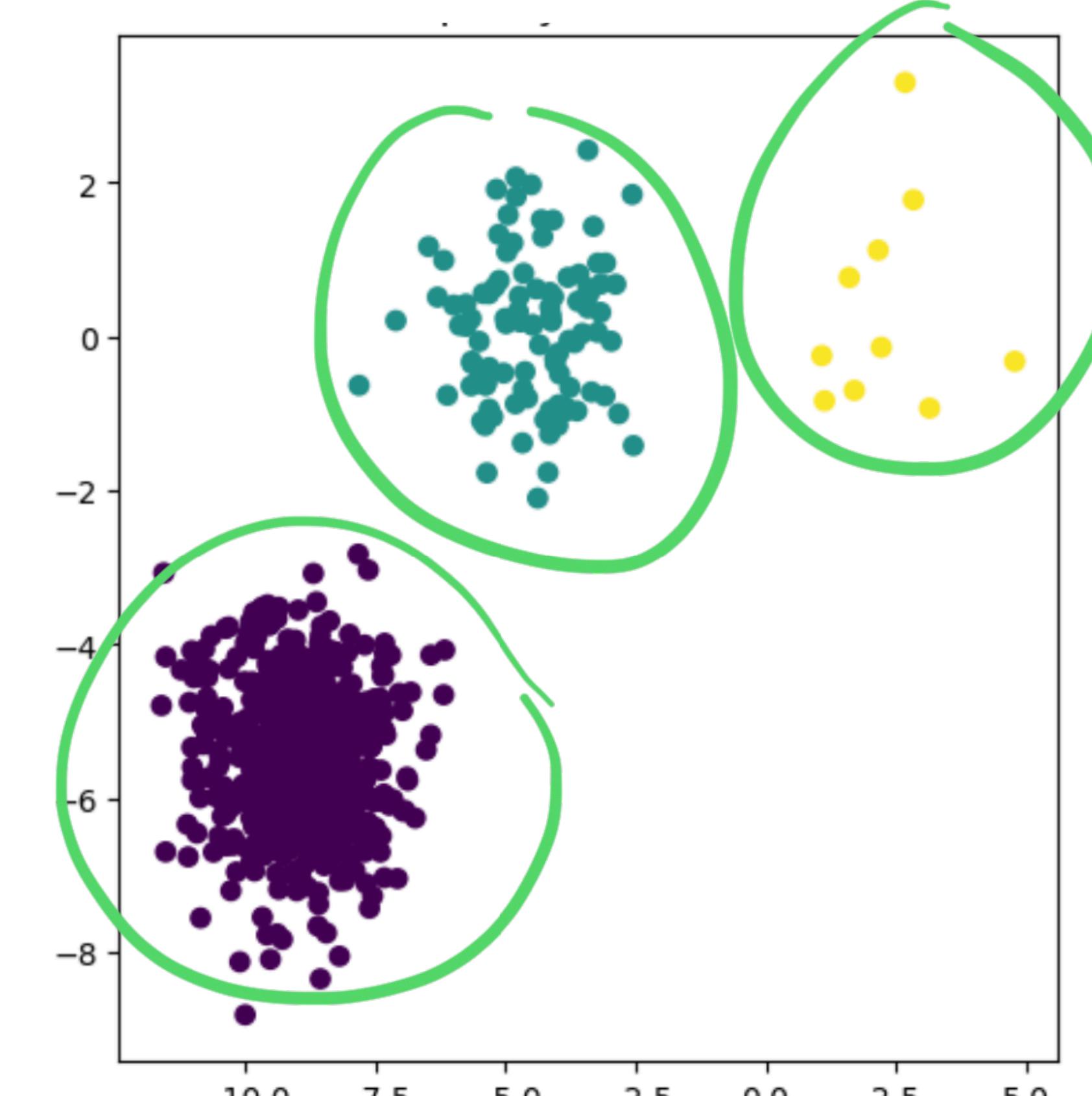


EM

# EM vs K-Means: Uneven Sizes



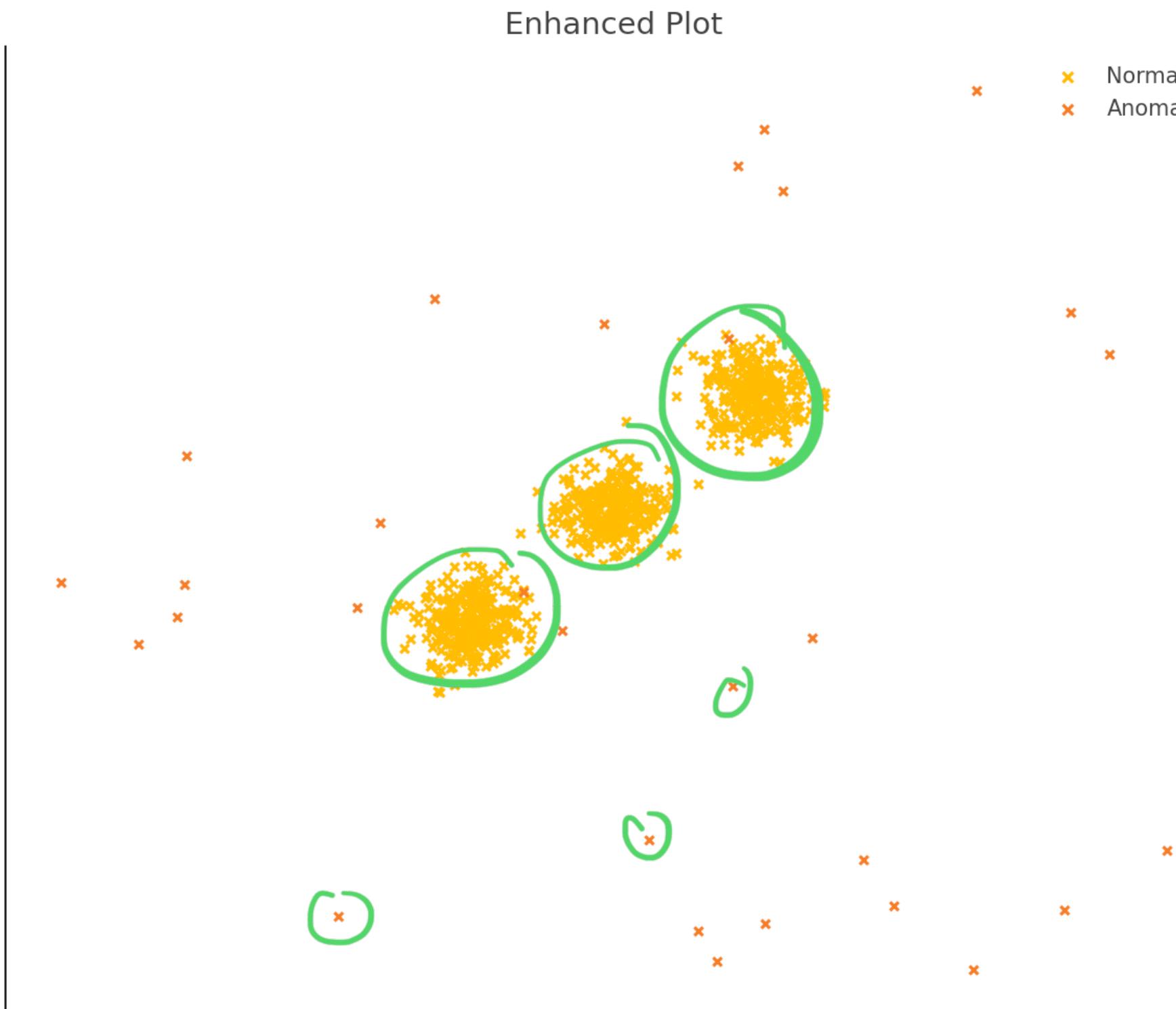
K-means



EM

# Anomaly Detection with GMMs

After training a GMM, we compute the likelihood  $p(\mathbf{x})$  for any new data point:



$$p(\mathbf{x}) = \sum_{k=1}^K \frac{\pi_k \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k)\right)}{(2\pi)^{d/2} |\boldsymbol{\Sigma}_k|^{1/2}}$$

If  $p(x) \leq \tau \implies \text{anomaly}$

Likelihood-based anomaly detection: red points are flagged as anomalies based on the GMM threshold

# EM Algorithm

## Maximum Likelihood from Incomplete Data via the *EM* Algorithm

By A. P. DEMPSTER, N. M. LAIRD and D. B. RUBIN

*Harvard University and Educational Testing Service*

[Read before the ROYAL STATISTICAL SOCIETY at a meeting organized by the RESEARCH SECTION on Wednesday, December 8th, 1976. Professor S. D. SILVEY in the Chair]

### SUMMARY

A broadly applicable algorithm for computing maximum likelihood estimates from incomplete data is presented at various levels of generality. Theory showing the monotone behaviour of the likelihood and convergence of the algorithm is derived. Many examples are sketched, including missing value situations, applications to grouped, censored or truncated data, finite mixture models, variance component estimation, hyperparameter estimation, iteratively reweighted least squares and factor analysis.

# EM Algorithm in General

**Assumption:**  $(\mathbf{x}, z)$  are random variables, where  $\mathbf{x}$  is observed (our data) and  $z$  is not observed (e.g. cluster membership)

Their joint distribution under a parametric model is given by  $p(\mathbf{x}, z | \theta)$

**Goal:** Given a dataset  $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_N)$ , maximize the log likelihood function defined in terms of the latent variables  $Z = (z_1, \dots, z_N)$ :

$$\begin{aligned}\underline{\mathcal{L}(\theta)} &= \log(p(\mathbf{X} | \theta)) = \log \left( \sum_Z p(\mathbf{X}, Z | \theta) \right) \\ &= \sum_{n=1}^N \log(p(\mathbf{x}_n | \theta)) = \sum_{n=1}^N \log \sum_{z_n} p(\mathbf{x}_n, z_n | \theta)\end{aligned}$$

$\log \prod_{n=1}^N p(\mathbf{x}_n | \theta)$

**Applications:** GMM, HMM and many others!

# EM Algorithm in General

We introduce a distribution  $q(Z)$  over the latent variables, i.e.  $q(Z) \geq 0$  and  $\sum_Z q(Z) = 1$ . For any such  $q(Z)$ ,

$$\begin{aligned}\mathcal{L}(\theta) &= \log(p(\mathbf{X} | \theta)) = \sum_Z q(Z) \log(p(\mathbf{X} | \theta)) \\ &= \sum_Z q(Z) \log \left( \frac{p(\mathbf{X} | \theta)p(Z | \mathbf{X}, \theta)q(Z)}{p(Z | \mathbf{X}, \theta)q(Z)} \right) \\ &= \sum_Z q(Z) \log \left( \frac{p(\mathbf{X}, Z | \theta)}{q(Z)} \right) + \sum_Z q(Z) \log \left( \frac{q(Z)}{p(Z | \mathbf{X}, \theta)} \right)\end{aligned}$$

*entropy*

$\therefore \mathcal{L}(q, \theta) := \mathcal{L}(q, \theta)$

$\therefore KL(q \| p) := KL(q \| p) \geq 0$

$= \sum_Z q(Z) \log \frac{q(Z)}{p(Z)}$

$= -\sum_Z q(Z) \log p(Z) - (-\sum_Z q(Z) \log q(Z))$

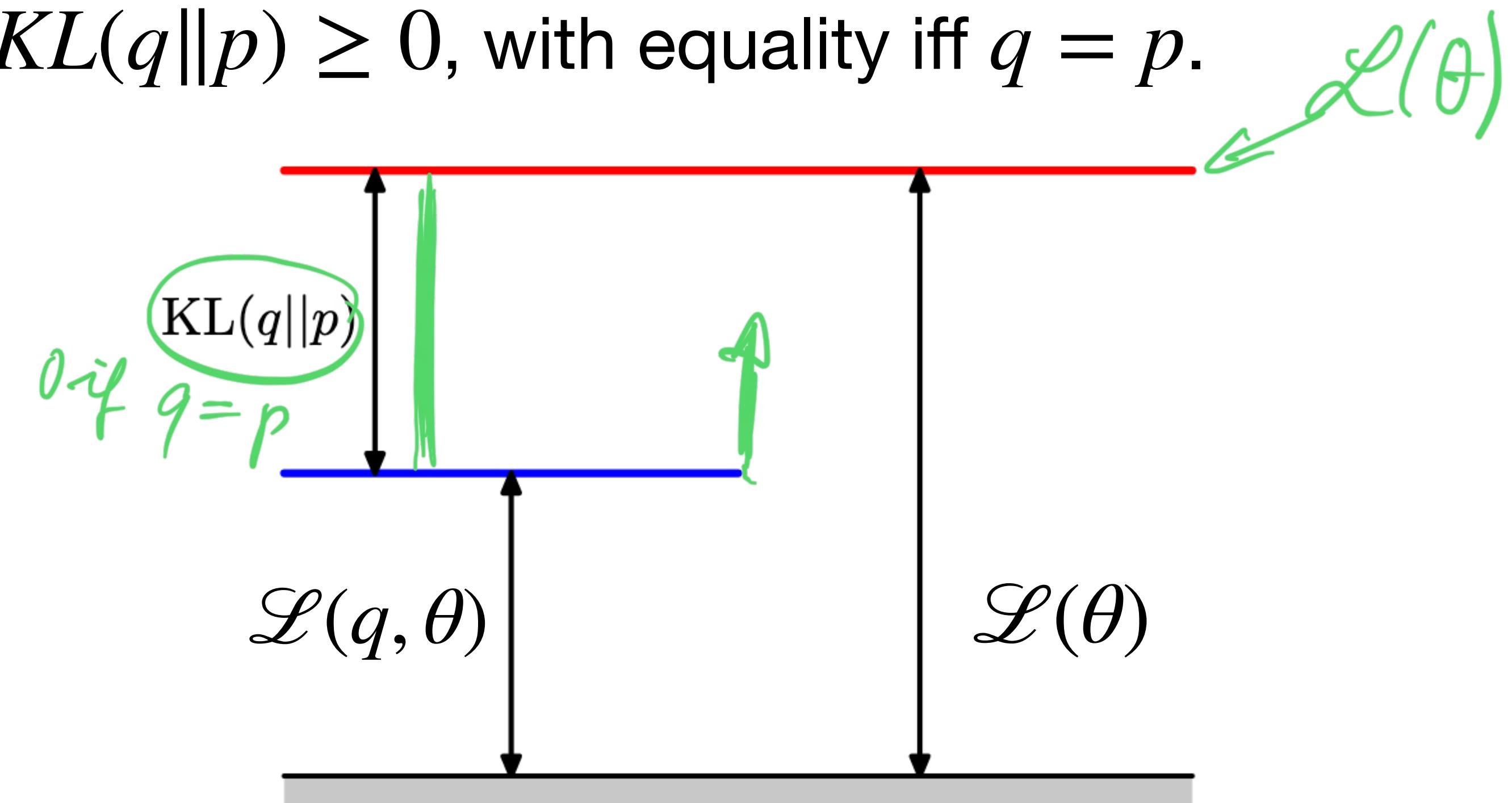
# EM Algorithm in General

$$\underline{\mathcal{L}(\theta)} = \mathcal{L}(q, \theta) + KL(q \| p) \geq 0$$

Note that for any  $q, p$  it holds that  $KL(q \| p) \geq 0$ , with equality iff  $q = p$ .

Therefore  $\forall q, \mathcal{L}(\theta) \geq \mathcal{L}(q, \theta)$

with equality iff  $q(Z) = p(Z | X, \theta)$



We have introduced an auxiliary function  $\mathcal{L}(q, \theta)$  that is always below the  $\mathcal{L}(\theta)$

# EM Algorithm as Maximizing $\mathcal{L}(q, \theta)$

The EM algorithm is a **coordinate ascent algorithm** on the function  $\mathcal{L}(q, \theta)$

E-step

$$q^{(t+1)} = \arg \max_q \mathcal{L}(q, \theta^{(t)})$$

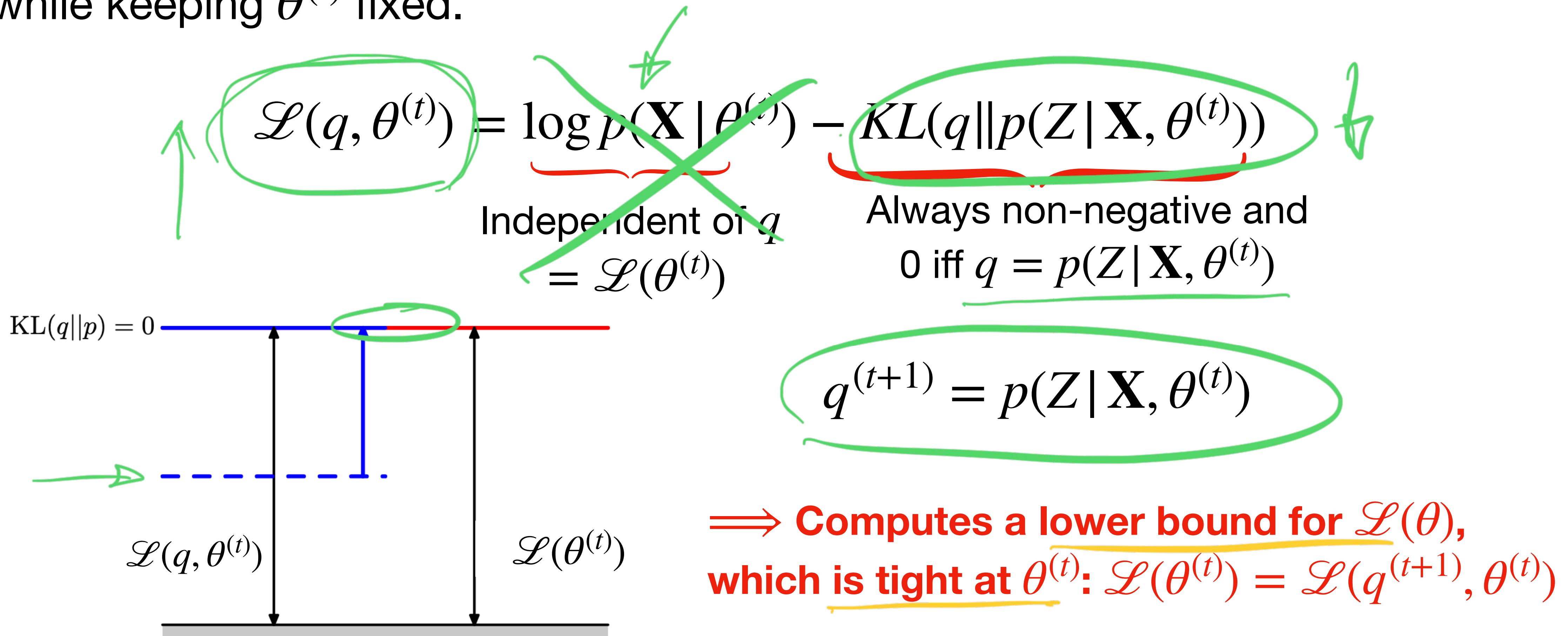
M-step

$$\theta^{(t+1)} = \arg \max_{\theta} \mathcal{L}(q^{(t+1)}, \theta)$$

$\mathcal{L}(\theta)$

# EM Algorithm as Maximizing $\mathcal{L}(q, \theta)$

The **E-step** of EM maximizes the lower bound  $\mathcal{L}(q, \theta^{(t)})$  with respect to  $q$ , while keeping  $\theta^{(t)}$  fixed.



# EM Algorithm as Maximizing $\mathcal{L}(q, \theta)$

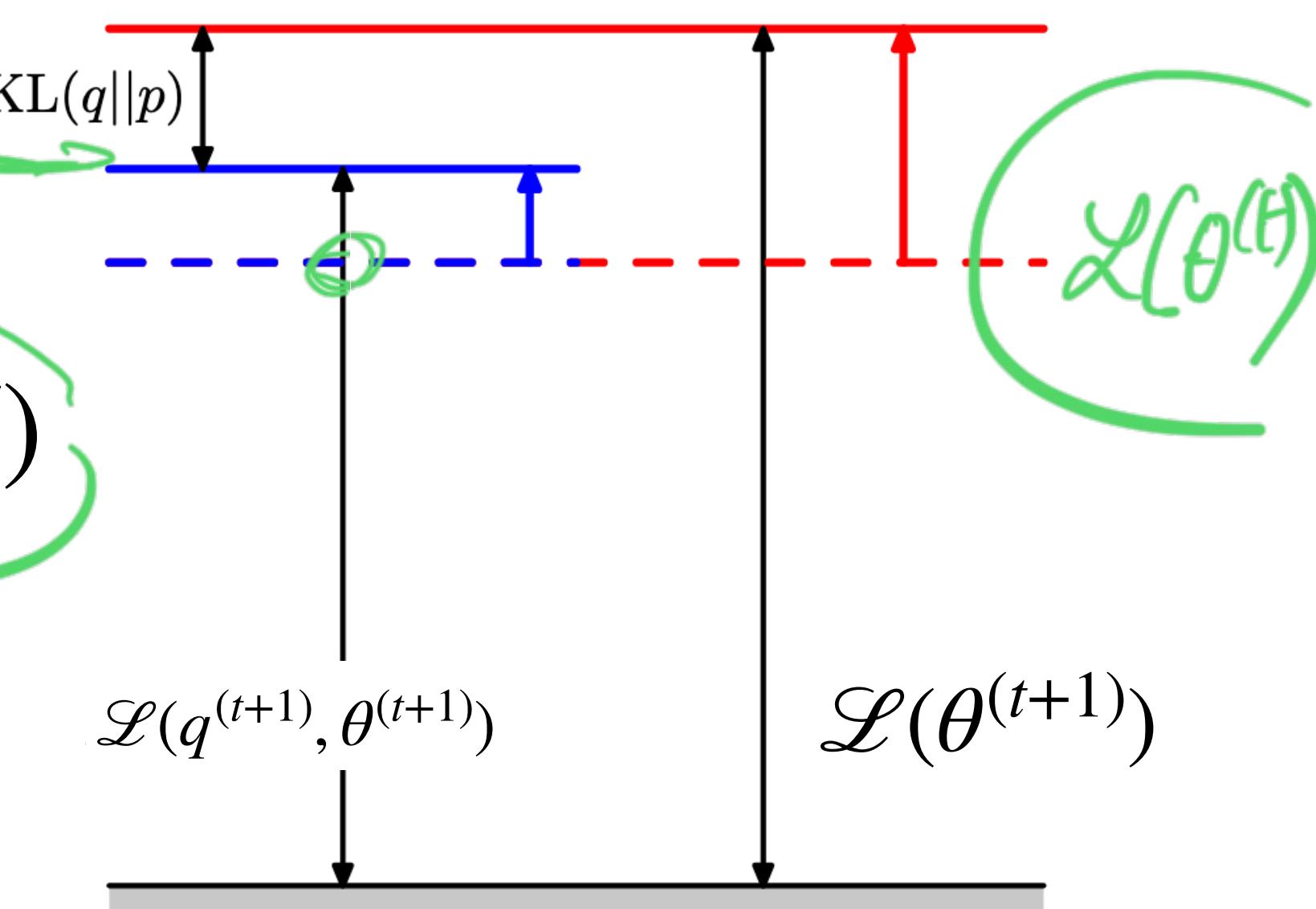
The **M-step** of EM maximizes the lower bound  $\mathcal{L}(q^{(t+1)}, \theta)$  with respect to  $\theta$ , while keeping  $q^{(t+1)}$  fixed:

Note that,

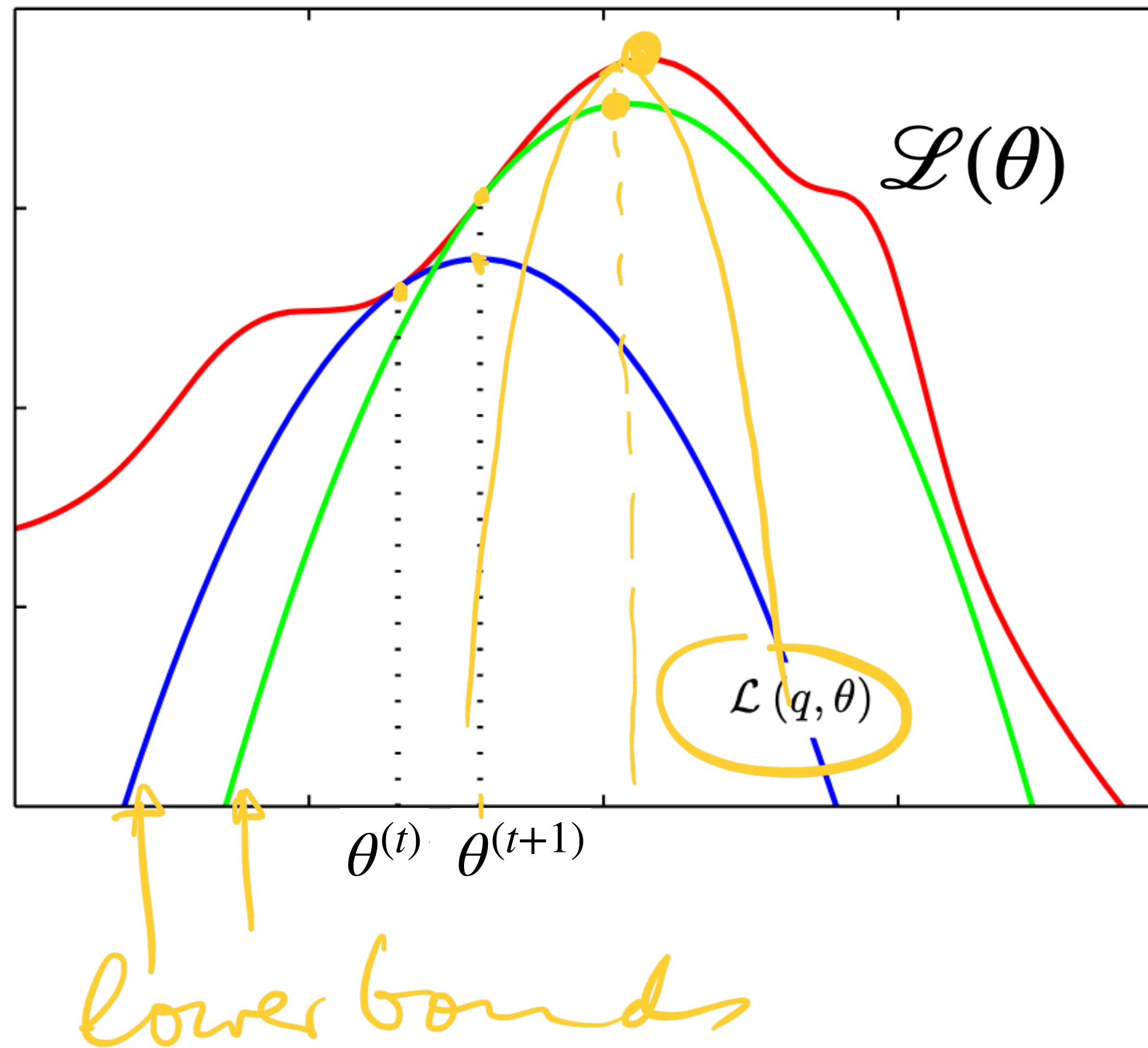
$$\begin{aligned}\mathcal{L}(q, \theta) &\stackrel{\Delta}{=} \sum_Z q(Z) \log \frac{p(\mathbf{X}, Z | \theta)}{q(Z)} \\ &= \sum_Z q(Z) \log p(\mathbf{X}, Z | \theta) - \sum_Z q(Z) \log q(Z) \\ &= \mathbb{E}_{Z \sim q} [\log p(\mathbf{X}, Z | \theta)] + H(q)\end{aligned}$$

Maximize the expected complete log likelihood

Independent of  $\theta$



# EM as an Algorithm Maximizing $\mathcal{L}(q, \theta)$



The EM algorithm involves alternately computing a lower bound on the log likelihood for the current parameter values (**E-step**) and then maximizing this bound to obtain the new parameter values (**M-step**).

# The EM Recipe

**Goal:** Maximize the incomplete likelihood  $\mathcal{L}(\theta) = \log(p(\mathbf{X} | \theta))$

**How:** Maximize its lower bound  $\mathcal{L}(q, \theta) = \log\left(\frac{p(\mathbf{X}, \mathbf{Z} | \theta)}{q(\mathbf{Z})}\right)$

## 1. E-step:

- Compute  $p(\mathbf{Z} | \mathbf{X}, \theta^{(t)})$  (corresponding to  $q^{(t+1)} = \arg \max_q \mathcal{L}(q, \theta^{(t)})$ )
- Write the complete likelihood  $\log p(\mathbf{X}, \mathbf{Z} | \theta)$
- Calculate the conditional expected value of the complete log likelihood  
 $\mathbb{E}_{\mathbf{Z}}[\log p(\mathbf{X}, \mathbf{Z} | \theta) | \mathbf{X}, \theta^{(t)}]$

## 2. M-step: Maximize $\theta^{(t+1)} = \arg \max_{\theta} \mathbb{E}_{\mathbf{Z}}[\log p(\mathbf{X}, \mathbf{Z} | \theta) | \mathbf{X}, \theta^{(t)}]$

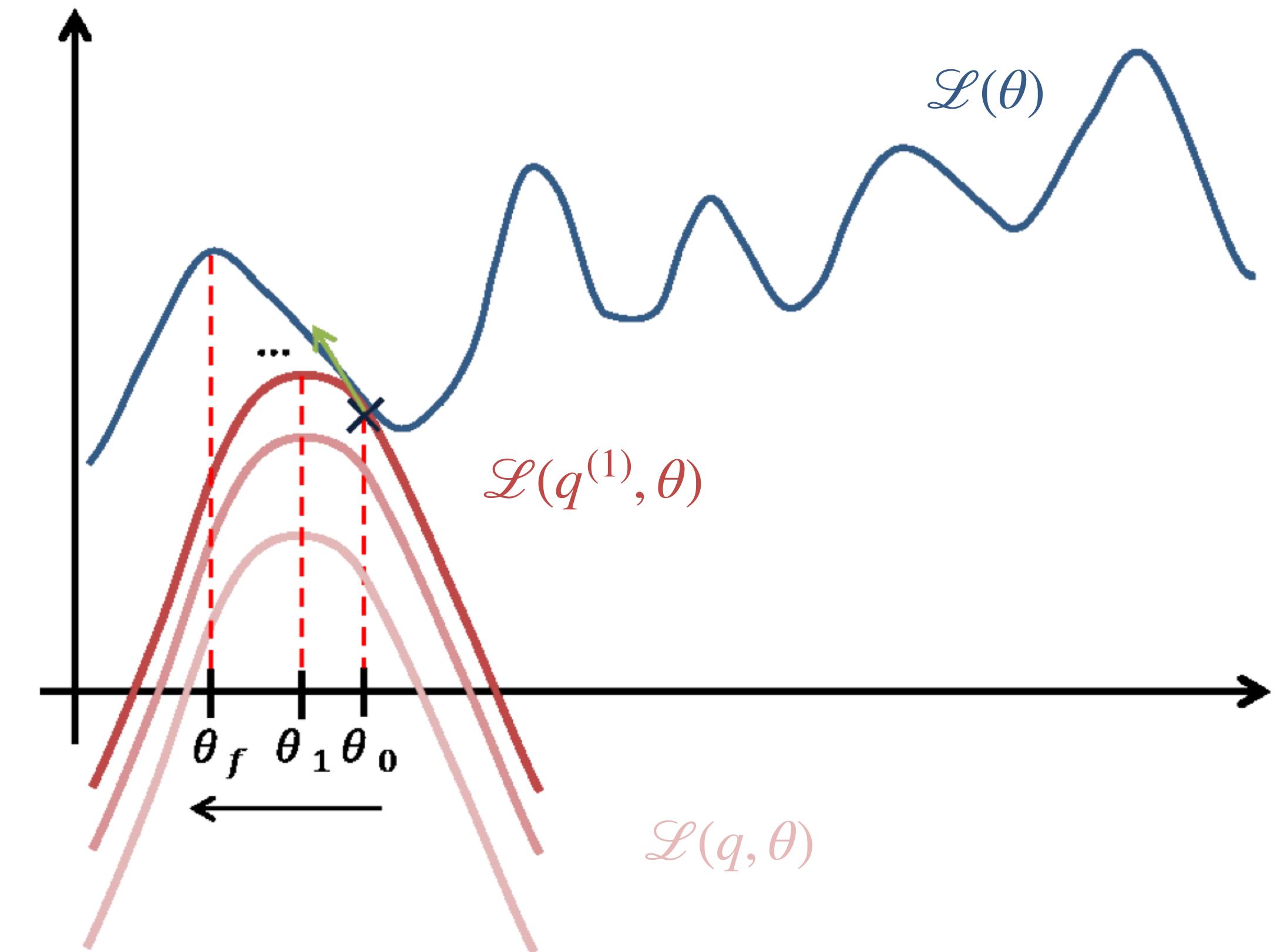
(corresponding to  $\theta^{(t+1)} = \arg \max_{\theta} \mathcal{L}(q^{(t+1)}, \theta)$ )

# EM convergence properties

- It is an ascent algorithm

$$\mathcal{L}(\theta^{(t)}) \leq \mathcal{L}(\theta^{(t+1)})$$

- The sequence of log-likelihoods converges (under suitable conditions)
- It does not converge to a global maximum but rather to a stationary point (non-convex problem)
- As with K-means, we reiterate the process to increase confidence and keep the one with the highest likelihood



# Practicalities: Selecting K

When  $K$  is too large ( $K = N$ ), placing one Gaussian at each data point ( $\mu_k = x_k$ ), shrinking variance ( $\Sigma_n \rightarrow 0_D$ ) maximizes the log-likelihood

$$\mathcal{L}(\theta) = - \sum_{n=1}^N \log((2\pi)^{D/2} |\Sigma_n|^{1/2}) - \log \left( \frac{1}{2}(x_n - \mu_n)^T \Sigma_n^{-1} (x_n - \mu_n) \right) \rightarrow \infty$$

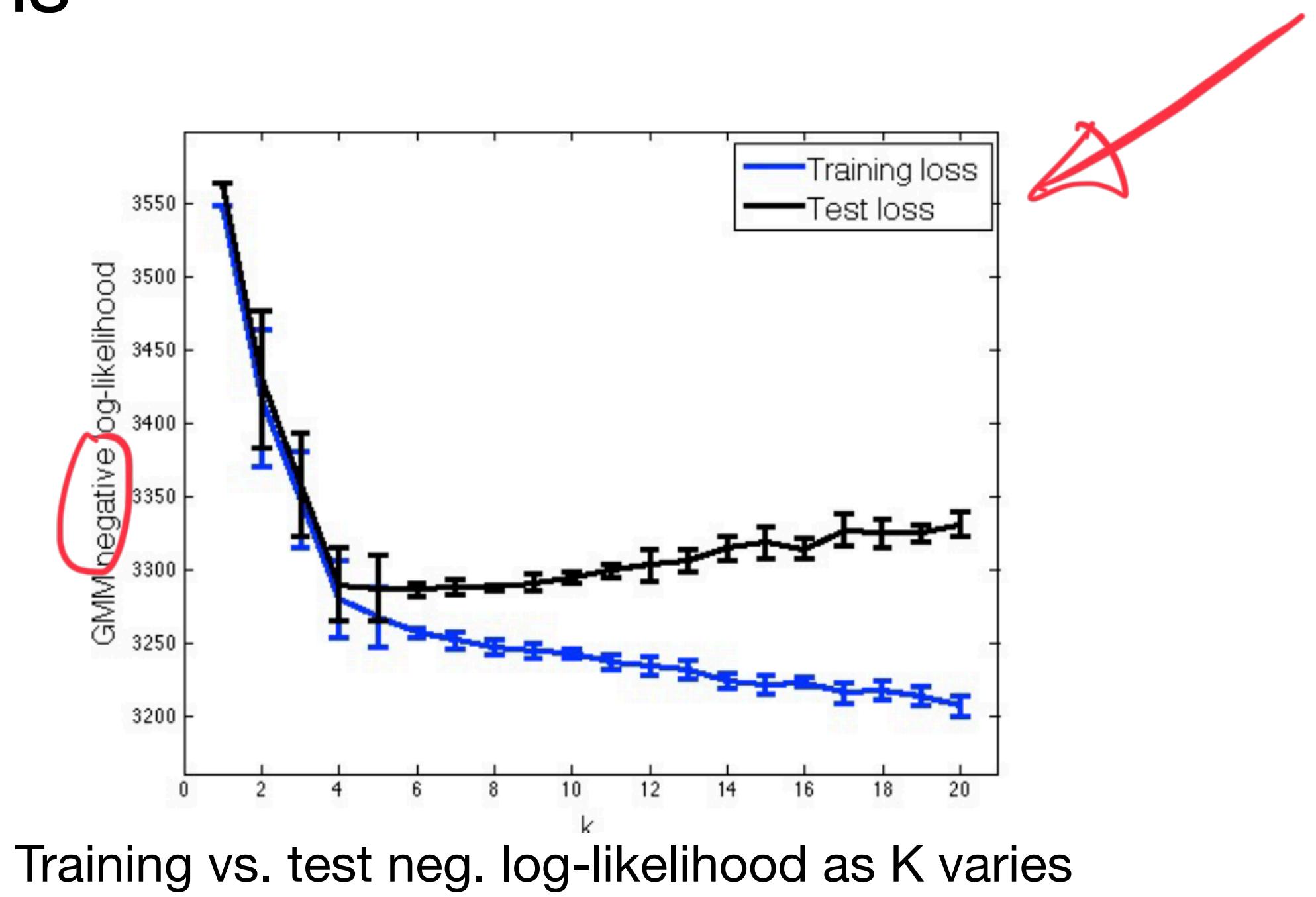
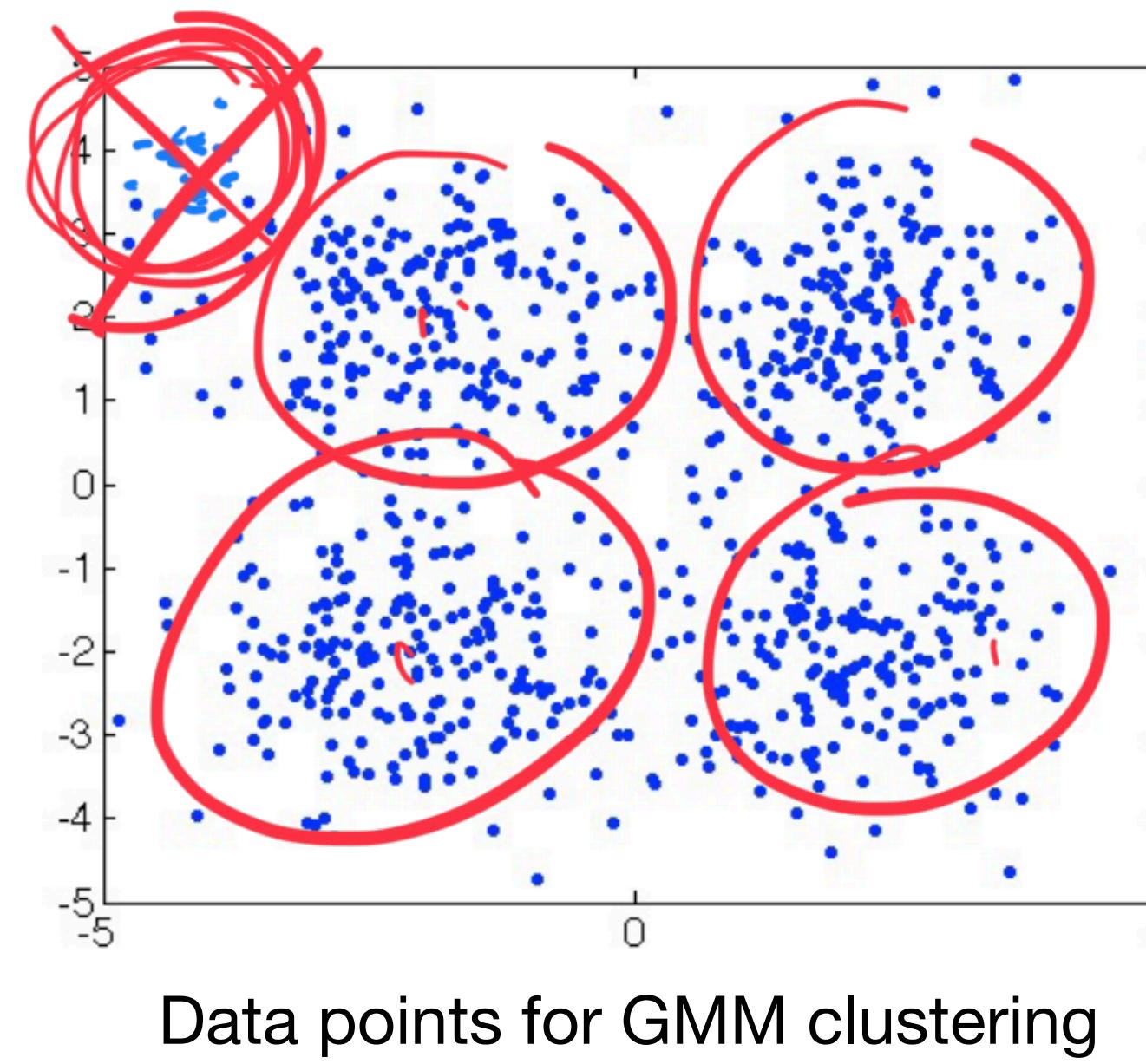
as  $\Sigma_n \rightarrow 0I_d$  and  $\mu_n = x_n$

Overfitting results in poor test set log-likelihood!

# Selecting K: Using Cross-Validation

Cross-validation selects  $K$  by maximizing the log-likelihood on validation dataset

This method was not possible for K-means



Cross-validation identifies the optimal  $K$  by maximizing test log-likelihood

# Practicalities: Initialization

For cluster weights:

- Typically use uniform distribution

For means:

- Randomly initialize
- Use K-means++

For covariance matrices:

- Initialize according to empirical variances in individual feature dimensions

$$\Sigma_1 = \dots = \Sigma_D = \begin{bmatrix} \sigma_1^2 & & \\ & \sigma_2^2 & \\ & & \sigma_K^2 \end{bmatrix}, \quad \sigma_d = \frac{1}{N} \sum_{n=1}^N (x_n[j] - \bar{x}_j)^2, \quad \bar{x}_j = \frac{1}{N} \sum_{n=1}^N x_n[j]$$

# Recap

- **Gaussian Mixture Models:**
  - Represents the data as a mixture of Gaussian distributions
  - Useful for clustering, density estimation, and anomaly detection
  - Offers more flexibility in cluster shape and size compared to K-means
- **Expectation Maximization Algorithm:**
  - Goal: Solve models with latent (hidden) variables (e.g. GMMs)
  - Process: Alternate between the E and M steps until convergence
    - E-step: Calculate the posterior of the latent variables == Compute expected complete log likelihood == Maximize  $\mathcal{L}(q, \theta)$  over  $q$
    - M-step: Maximize the expected complete log likelihood == Maximize  $\mathcal{L}(q, \theta)$  over  $\theta$