Lecture 8: Introduction to Unsupervised Learning, K-means, Gaussian Mixture Model (GMM), and EM algorithm

Tao LIN

SoE, Westlake University

November 4, 2025

Table of Contents

Table of Contents

Motivation of Expectation-Maximization (EM)

Recall the equation of maximum likelihood for GMM:

$$\max_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) := \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{1}$$

Motivation of Expectation-Maximization (EM)

Recall the equation of maximum likelihood for GMM:

$$\max_{\boldsymbol{\theta}} \ \mathcal{L}(\boldsymbol{\theta}) := \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{1}$$

Remarks:

Motivation of Expectation-Maximization (EM)

Recall the equation of maximum likelihood for GMM:

$$\max_{\boldsymbol{\theta}} \; \mathcal{L}(\boldsymbol{\theta}) := \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{1}$$

Remarks:

- Computing maximum likelihood for GMM is difficult, due to .

Motivation of Expectation-Maximization (EM)

Recall the equation of maximum likelihood for GMM:

$$\max_{\boldsymbol{\theta}} \; \mathcal{L}(\boldsymbol{\theta}) := \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{1}$$

Remarks:

- Computing maximum likelihood for GMM is difficult, due to the $\log$ outside the sum.

EM provides an elegant solution by iteratively building & optimizing lower bounds.

Motivation of Expectation-Maximization (EM)

Recall the equation of maximum likelihood for GMM:

$$\max_{\boldsymbol{\theta}} \; \mathcal{L}(\boldsymbol{\theta}) := \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{1}$$

Remarks:

- Computing maximum likelihood for GMM is difficult, due to the $\log$ outside the sum.

EM provides an elegant solution by iteratively building & optimizing lower bounds.

- It uses an iterative two-step procedure, where individual steps usually involve problems that are easy to optimize.

An Overview of EM

- The EM algorithm provides an elegant approach to latent variable estimation problems.
- EM iteratively tries to overcome the challenge of not knowing the hidden variables $\boldsymbol{z}_n$.

An Overview of EM

- The EM algorithm provides an elegant approach to latent variable estimation problems.
- EM iteratively tries to overcome the challenge of not knowing the hidden variables $\mathbf{z}_n$.
- If we knew the values of the latent variables $\mathbf{z}_n$, maximum likelihood estimation would be simple.

An Overview of EM

- The EM algorithm provides an elegant approach to latent variable estimation problems.
- EM iteratively tries to overcome the challenge of not knowing the hidden variables $\mathbf{z}_n$.
- If we knew the values of the latent variables $\mathbf{z}_n$, maximum likelihood estimation would be simple.
- Since we don't know them, we:

An Overview of EM

- The EM algorithm provides an elegant approach to latent variable estimation problems.
- EM iteratively tries to overcome the challenge of not knowing the hidden variables $\mathbf{z}_n$.
- If we knew the values of the latent variables $\mathbf{z}_n$, maximum likelihood estimation would be simple.
- Since we don't know them, we:
  - **E-step:** Compute the *distribution* of the latent variables given our current parameters.

An Overview of EM

- The EM algorithm provides an elegant approach to latent variable estimation problems.
- EM iteratively tries to overcome the challenge of not knowing the hidden variables $\mathbf{z}_n$.
- If we knew the values of the latent variables $\mathbf{z}_n$, maximum likelihood estimation would be simple.
- Since we don't know them, we:
    - **E-step:** Compute the *distribution* of the latent variables given our current parameters.
    - **M-step:** Update the parameters using these *expected* latent variable values.

A Note on Notation: z_n (scalar) vs. $\mathbf{z}_n$ (vector)

The latent variable for $\mathbf{x}_n$ can be written in two ways:

- $\mathbf{z}_n$ is the **one-hot vector** (formal notation).
  - A K-dimensional vector, e.g., $\mathbf{z}_n = [0, 1, 0, \dots, 0]^\top$.
  - Useful for writing the complete-data likelihood compactly:

$$p(\mathbf{x}_n, \mathbf{z}_n | \boldsymbol{\theta}) = \prod_{k=1}^{K} [\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)]^{z_{nk}}$$

- z_n is the **scalar index** (convenient notation).
  - A single number, e.g., $z_n = 2$.
  - $z_n = k$ is equivalent to the vector $\mathbf{z}_n$ having $z_{nk} = 1$.

- $q_n(z_n)$ is our **approximating posterior distribution**.
  - Since z_n takes one of K discrete values, $q_n(z_n)$ is a *categorical distribution*.
  - In the E-step, we set it to the true posterior: $q_n(z_n) = p(z_n \,|\, \mathbf{x}_n, \boldsymbol{\theta}^{(t)})$.

- $q_{kn}^{(t)}$ is the **probability** of component k for point n. This single value connects all notations:

$$q_{kn}^{(t)} := q_n(z_n = k) = p(z_n = k \,|\, \mathbf{x}_n, \boldsymbol{\theta}^{(t)}) = \mathbb{E}_{q_n}[\mathbf{z}_{nk}]$$

EM algorithm: Formal Summary

Start with $\boldsymbol{\theta}^{(1)}$ and iterate:

EM algorithm: Formal Summary

Start with $\boldsymbol{\theta}^{(1)}$ and iterate:

1. Expectation step: Compute a lower bound $\underline{\mathcal{L}}$ to the true log-likelihood $\mathcal{L}$ such that it is *tight* at the current parameters $\boldsymbol{\theta}^{(t)}$:

$$\mathcal{L}(\boldsymbol{\theta}) \geq \underline{\mathcal{L}}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) \text{ and} \tag{2}$$

$$\mathcal{L}(\boldsymbol{\theta}^{(t)}) = \underline{\mathcal{L}}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t)}) \,. \tag{3}$$

EM algorithm: Formal Summary

Start with $\boldsymbol{\theta}^{(1)}$ and iterate:

1. Expectation step: Compute a lower bound $\underline{\mathcal{L}}$ to the true log-likelihood $\mathcal{L}$ such that it is *tight* at the current parameters $\boldsymbol{\theta}^{(t)}$:

$$\mathcal{L}(\boldsymbol{\theta}) \geq \underline{\mathcal{L}}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) \text{ and} \tag{2}$$

$$\mathcal{L}(\boldsymbol{\theta}^{(t)}) = \underline{\mathcal{L}}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t)}) . \tag{3}$$

2. Maximization step: Find the new parameters $\boldsymbol{\theta}^{(t+1)}$ by maximizing this lower bound:

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \underline{\mathcal{L}}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) . \tag{4}$$

EM algorithm: Formal Summary

Start with $\boldsymbol{\theta}^{(1)}$ and iterate:

1. Expectation step: Compute a lower bound $\underline{\mathcal{L}}$ to the true log-likelihood $\mathcal{L}$ such that it is *tight* at the current parameters $\boldsymbol{\theta}^{(t)}$:

$$\mathcal{L}(\boldsymbol{\theta}) \geq \underline{\mathcal{L}}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) \text{ and} \tag{2}$$

$$\mathcal{L}(\boldsymbol{\theta}^{(t)}) = \underline{\mathcal{L}}(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t)}) \,. \tag{3}$$

2. Maximization step: Find the new parameters $\boldsymbol{\theta}^{(t+1)}$ by maximizing this lower bound:

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \underline{\mathcal{L}}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) \,. \tag{4}$$

Each iteration of EM is guaranteed to increase (or keep the same) the true log-likelihood $\mathcal{L}(\boldsymbol{\theta})$.

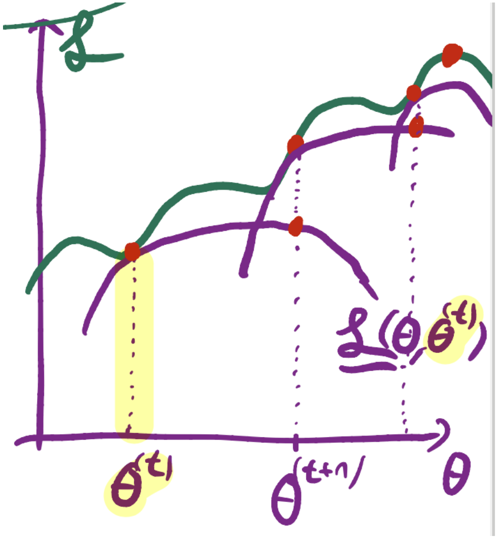


Figure: Visualization of EM iterations: Each M-step maximizes a new lower bound $\underline{\mathcal{L}}$ (purple), guaranteeing an increase in the true log-likelihood $\mathcal{L}$ (green).

Background: What is KL Divergence?

The Kullback-Leibler (KL) Divergence is a measure of how one probability distribution $q(z)$ is different from a second, reference probability distribution $p(z)$.

For discrete distributions:

$$\mathrm{KL}(q \parallel p) = \sum_z q(z) \log \frac{q(z)}{p(z)}$$

Key Properties:

Background: What is KL Divergence?

The Kullback-Leibler (KL) Divergence is a measure of how one probability distribution $q(z)$ is different from a second, reference probability distribution $p(z)$.

For discrete distributions:

$$\mathrm{KL}(q \parallel p) = \sum_z q(z) \log \frac{q(z)}{p(z)}$$

Key Properties:

- **It is not a true "distance":**
    - It is not symmetric: $\mathrm{KL}(q \parallel p) \neq \mathrm{KL}(p \parallel q)$

Background: What is KL Divergence?

The Kullback-Leibler (KL) Divergence is a measure of how one probability distribution $q(z)$ is different from a second, reference probability distribution $p(z)$.

For discrete distributions:

$$\mathrm{KL}(q \parallel p) = \sum_z q(z) \log \frac{q(z)}{p(z)}$$

Key Properties:

- **It is not a true "distance":**
  - It is not symmetric: $\mathrm{KL}(q \parallel p) \neq \mathrm{KL}(p \parallel q)$
- **It is always non-negative:**
  - $\mathrm{KL}(q \parallel p) \geq 0$

Background: What is KL Divergence?

The Kullback-Leibler (KL) Divergence is a measure of how one probability distribution $q(z)$ is different from a second, reference probability distribution $p(z)$.

For discrete distributions:

$$\text{KL}(q \parallel p) = \sum_z q(z) \log \frac{q(z)}{p(z)}$$

Key Properties:

- **It is not a true "distance":**
  - It is not symmetric: $\text{KL}(q \parallel p) \neq \text{KL}(p \parallel q)$
- **It is always non-negative:**
  - $\text{KL}(q \parallel p) \geq 0$
- **It is zero only if the distributions are identical:**
  - $\text{KL}(q \parallel p) = 0 \Longleftrightarrow q(z) = p(z)$ for all z.

Background: What is KL Divergence?

The Kullback-Leibler (KL) Divergence is a measure of how one probability distribution $q(z)$ is different from a second, reference probability distribution $p(z)$.

For discrete distributions:

$$\mathrm{KL}(q \parallel p) = \sum_z q(z) \log \frac{q(z)}{p(z)}$$

Key Properties:

- **It is not a true "distance":**
  - It is not symmetric: $\mathrm{KL}(q \parallel p) \neq \mathrm{KL}(p \parallel q)$
- **It is always non-negative:**
  - $\mathrm{KL}(q \parallel p) \geq 0$
- **It is zero only if the distributions are identical:**
  - $\mathrm{KL}(q \parallel p) = 0 \Longleftrightarrow q(z) = p(z)$ for all z.

In EM, we use KL divergence to measure how "bad" our approximation $q(\mathbf{z})$ is compared to the true posterior $p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})$.

The Evidence Lower Bound (ELBO)

- This lower bound is known as the Evidence Lower Bound (ELBO).

The Evidence Lower Bound (ELBO)

- This lower bound is known as the Evidence Lower Bound (ELBO).
- The log-likelihood (evidence) can be decomposed for *any* distribution $q(\mathbf{z})$:

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + \mathsf{KL}(q(\mathbf{z}) \parallel p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \tag{5}$$

where $\mathcal{L}(q, \boldsymbol{\theta})$ is the ELBO:

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})] - \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] \tag{6}$$

The Evidence Lower Bound (ELBO)

- This lower bound is known as the Evidence Lower Bound (ELBO).
- The log-likelihood (evidence) can be decomposed for *any* distribution $q(\mathbf{z})$:

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + \mathrm{KL}(q(\mathbf{z}) \parallel p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \tag{5}$$

  where $\mathcal{L}(q, \boldsymbol{\theta})$ is the ELBO:

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})] - \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] \tag{6}$$

- Since $\mathrm{KL}(\cdot) \geq 0$, the ELBO $\mathcal{L}(q, \boldsymbol{\theta})$ is always a *lower bound* on the true log-likelihood.

The Evidence Lower Bound (ELBO)

- This lower bound is known as the Evidence Lower Bound (ELBO).
- The log-likelihood (evidence) can be decomposed for *any* distribution $q(\mathbf{z})$:

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + \mathrm{KL}(q(\mathbf{z}) \parallel p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \tag{5}$$

  where $\mathcal{L}(q, \boldsymbol{\theta})$ is the ELBO:

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})] - \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] \tag{6}$$

- Since $\mathrm{KL}(\cdot) \geq 0$, the ELBO $\mathcal{L}(q, \boldsymbol{\theta})$ is always a *lower bound* on the true log-likelihood.
- EM alternates between:
  - **E-step**: Maximize $\mathcal{L}$ w.r.t. q. This happens when $\mathrm{KL} = 0$, so we set $q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^{(t)})$.
  - **M-step**: Maximize $\mathcal{L}$ w.r.t. $\boldsymbol{\theta}$, keeping q fixed.

ELBO: Intuition and Properties

- The ELBO can be rewritten in two equivalent forms:

$$\begin{aligned} \mathcal{L}(q, \boldsymbol{\theta}) &= \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})] - \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] \qquad (7) \\ &= \log p(\mathbf{x}|\boldsymbol{\theta}) - \mathsf{KL}(q(\mathbf{z}) \parallel p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \qquad (8) \end{aligned}$$

ELBO: Intuition and Properties

- The ELBO can be rewritten in two equivalent forms:

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})] - \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] \tag{7}$$

$$= \log p(\mathbf{x}|\boldsymbol{\theta}) - \mathrm{KL}(q(\mathbf{z}) \parallel p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \tag{8}$$

- The first form (7) shows ELBO as:
  - Expected complete-data log-likelihood (tries to fit data)
  - Minus entropy of q (tries to keep q “spread out”)

ELBO: Intuition and Properties

- The ELBO can be rewritten in two equivalent forms:

$$\begin{aligned} \mathcal{L}(q, \boldsymbol{\theta}) &= \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})] - \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] && (7) \\ &= \log p(\mathbf{x}|\boldsymbol{\theta}) - \mathrm{KL}(q(\mathbf{z}) \parallel p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) && (8) \end{aligned}$$

- The first form (7) shows ELBO as:
  - Expected complete-data log-likelihood (tries to fit data)
  - Minus entropy of q (tries to keep q “spread out”)
- The second form (8) shows ELBO as:
  - The true log-likelihood (what we want)
  - Minus the KL divergence (how bad our q is)

ELBO: Intuition and Properties

- The ELBO can be rewritten in two equivalent forms:

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})] - \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] \quad (7)$$

$$= \log p(\mathbf{x}|\boldsymbol{\theta}) - \mathrm{KL}(q(\mathbf{z}) \parallel p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \quad (8)$$

- The first form (7) shows ELBO as:
  - Expected complete-data log-likelihood (tries to fit data)
  - Minus entropy of q (tries to keep q “spread out”)
- The second form (8) shows ELBO as:
  - The true log-likelihood (what we want)
  - Minus the KL divergence (how bad our q is)
- EM optimization balances:
  - Finding a good fit to the data through $\boldsymbol{\theta}$.
  - Finding a good approximation to the posterior through q.

The Expectation step (E-step) in Practice

Goal: Set $q_n(z_n) = p(z_n \mid \mathbf{x}_n, \boldsymbol{\theta}^{(t)})$.

The Expectation step (E-step) in Practice

Goal: Set $q_n(z_n) = p(z_n \mid \mathbf{x}_n, \boldsymbol{\theta}^{(t)})$.

How? We just need to find the probability for each component k:

$$\begin{aligned} q_{kn}^{(t)} &:= q_n(z_n = k) = p(z_n = k \mid \mathbf{x}_n, \boldsymbol{\theta}^{(t)}) \\ &= \frac{p(\mathbf{x}_n | z_n = k, \boldsymbol{\theta}^{(t)}) p(z_n = k | \boldsymbol{\theta}^{(t)})}{p(\mathbf{x}_n | \boldsymbol{\theta}^{(t)})} \quad \text{(Bayes' rule)} \\ &= \frac{p(\mathbf{x}_n | z_n = k, \boldsymbol{\theta}^{(t)}) p(z_n = k | \boldsymbol{\theta}^{(t)})}{\sum_{j=1}^{K} p(\mathbf{x}_n | z_n = j, \boldsymbol{\theta}^{(t)}) p(z_n = j | \boldsymbol{\theta}^{(t)})} \end{aligned}$$

The Expectation step (E-step) in Practice

Goal: Set $q_n(z_n) = p(z_n \mid \mathbf{x}_n, \boldsymbol{\theta}^{(t)})$.

How? We just need to find the probability for each component k:

$$\begin{aligned} q_{kn}^{(t)} &:= q_n(z_n = k) = p(z_n = k \mid \mathbf{x}_n, \boldsymbol{\theta}^{(t)}) \\ &= \frac{p(\mathbf{x}_n | z_n = k, \boldsymbol{\theta}^{(t)}) p(z_n = k | \boldsymbol{\theta}^{(t)})}{p(\mathbf{x}_n | \boldsymbol{\theta}^{(t)})} \quad \text{(Bayes' rule)} \\ &= \frac{p(\mathbf{x}_n | z_n = k, \boldsymbol{\theta}^{(t)}) p(z_n = k | \boldsymbol{\theta}^{(t)})}{\sum_{j=1}^{K} p(\mathbf{x}_n | z_n = j, \boldsymbol{\theta}^{(t)}) p(z_n = j | \boldsymbol{\theta}^{(t)})} \end{aligned}$$

This gives the famous E-step update rule:

$$q_{kn}^{(t)} = \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{j=1}^{K} \pi_j^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j^{(t)}, \boldsymbol{\Sigma}_j^{(t)})} \tag{9}$$

This is the “soft assignment” or “responsibility” of component k for point n.

The Maximization step (M-step)

Goal: Maximize the lower bound w.r.t. $\boldsymbol{\theta}$, using the $q_{kn}^{(t)}$ we just found.

$$\max_{\boldsymbol{\theta}} \sum_{n=1}^{N} \sum_{k=1}^{K} q_{kn}^{(t)} \left[\log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right] \tag{10}$$

The Maximization step (M-step)

Goal: Maximize the lower bound w.r.t. $\boldsymbol{\theta}$, using the $q_{kn}^{(t)}$ we just found.

$$\max_{\boldsymbol{\theta}} \sum_{n=1}^{N} \sum_{k=1}^{K} q_{kn}^{(t)} \left[\log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right] \tag{10}$$

This maximization can be solved independently for $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, and π_k. This results in intuitive “weighted MLE” updates:

$$\boldsymbol{\mu}_k^{(t+1)} := \frac{\sum_{n=1}^{N} q_{kn}^{(t)} \mathbf{x}_n}{\sum_{n=1}^{N} q_{kn}^{(t)}} \tag{11}$$

$$\boldsymbol{\Sigma}_k^{(t+1)} := \frac{\sum_{n=1}^{N} q_{kn}^{(t)} (\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})(\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})^\top}{\sum_{n=1}^{N} q_{kn}^{(t)}} \tag{12}$$

The Maximization step (M-step)

Goal: Maximize the lower bound w.r.t. $\boldsymbol{\theta}$, using the $q_{kn}^{(t)}$ we just found.

$$\max_{\boldsymbol{\theta}} \sum_{n=1}^{N} \sum_{k=1}^{K} q_{kn}^{(t)} \left[\log \pi_k + \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right] \tag{10}$$

This maximization can be solved independently for $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, and π_k. This results in intuitive "weighted MLE" updates:

$$\boldsymbol{\mu}_k^{(t+1)} := \frac{\sum_{n=1}^{N} q_{kn}^{(t)} \mathbf{x}_n}{\sum_{n=1}^{N} q_{kn}^{(t)}} \tag{11}$$

$$\boldsymbol{\Sigma}_k^{(t+1)} := \frac{\sum_{n=1}^{N} q_{kn}^{(t)} (\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})(\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})^\top}{\sum_{n=1}^{N} q_{kn}^{(t)}} \tag{12}$$

For π_k, we maximize (10) with a Lagrangian for the constraint $\sum_k \pi_k = 1$:

$$\pi_k^{(t+1)} := \frac{1}{N} \sum_{n=1}^{N} q_{kn}^{(t)} \tag{13}$$

(See derivation slides for full proof.)

M-step: The Objective Function (Why Eq. 10?)

The M-step maximizes the ELBO w.r.t. $\boldsymbol{\theta}$. From our ELBO definition:

$$\mathcal{L}(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})] - \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})]$$

Since the M-step optimizes for $\boldsymbol{\theta}$, we can ignore the second term, $\mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})]$, as it does not depend on $\boldsymbol{\theta}$. We are left with maximizing the expected complete-data log-likelihood:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) = \mathbb{E}_{q(\mathbf{z})}[\log p(\mathbf{X}, \mathbf{z}|\boldsymbol{\theta})] = \sum_{n=1}^{N} \mathbb{E}_{q_n(\mathbf{z}_n)}[\log p(\mathbf{x}_n, \mathbf{z}_n|\boldsymbol{\theta})]$$

We expand $\log p(\mathbf{x}_n, \mathbf{z}_n|\boldsymbol{\theta}) = \log p(\mathbf{x}_n|\mathbf{z}_n, \boldsymbol{\theta}) + \log p(z_n|\boldsymbol{\theta})$.

- $p(z_n = k|\boldsymbol{\theta}) = \pi_k$
- $p(\mathbf{x}_n|z_n = k, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

Using $q_{kn}^{(t)} := q_n(z_n = k)$ as the expectation weight, we get:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) = \sum_{n=1}^{N} \sum_{k=1}^{K} q_{kn}^{(t)} \left[\log \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) + \log \pi_k\right]$$

This is exactly the objective from the previous slide.

M-step Derivation: Mean $\boldsymbol{\mu}_k$

Goal: Find $\boldsymbol{\mu}_k$ that maximizes the objective. We only need terms that depend on $\boldsymbol{\mu}_k$:

$$\mathcal{L}(\boldsymbol{\mu}_k) = \sum_{n=1}^{N} q_{kn}^{(t)} \log \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) + \text{const} = \sum_{n=1}^{N} q_{kn}^{(t)} \left[\text{const} - \frac{1}{2}(\mathbf{x}_n - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k)\right]$$

Take the gradient w.r.t. $\boldsymbol{\mu}_k$ and set to 0 (using $\nabla_{\mathbf{z}}(\mathbf{a}-\mathbf{z})^\top \mathbf{C}(\mathbf{a}-\mathbf{z}) = 2\mathbf{C}(\mathbf{z}-\mathbf{a})$):

$$\nabla_{\boldsymbol{\mu}_k}\mathcal{L} = \sum_{n=1}^{N} q_{kn}^{(t)} \left[-\frac{1}{2}\cdot 2\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{\mu}_k - \mathbf{x}_n)\right] = \sum_{n=1}^{N} q_{kn}^{(t)} \left[\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k)\right] \overset{\text{set}}{=} 0$$

Multiply by $\boldsymbol{\Sigma}_k$ (which is invertible) to simplify:

$$\sum_{n=1}^{N} q_{kn}^{(t)}(\mathbf{x}_n - \boldsymbol{\mu}_k) = 0\,, \qquad \sum_{n=1}^{N} q_{kn}^{(t)}\mathbf{x}_n = \sum_{n=1}^{N} q_{kn}^{(t)}\boldsymbol{\mu}_k\,, \qquad \sum_{n=1}^{N} q_{kn}^{(t)}\mathbf{x}_n = \boldsymbol{\mu}_k\left(\sum_{n=1}^{N} q_{kn}^{(t)}\right),$$

$$\implies \boldsymbol{\mu}_k^{(t+1)} = \frac{\sum_{n=1}^{N} q_{kn}^{(t)}\mathbf{x}_n}{\sum_{n=1}^{N} q_{kn}^{(t)}}$$

M-step Derivation: Covariance $\boldsymbol{\Sigma}_k$

Goal: Find $\boldsymbol{\Sigma}_k$ that maximizes the objective. We only need terms that depend on $\boldsymbol{\Sigma}_k$:

$$\mathcal{L}(\boldsymbol{\Sigma}_k) = \sum_{n=1}^{N} q_{kn}^{(t)} \log \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) + \text{const} = \sum_{n=1}^{N} q_{kn}^{(t)} \left[\log \left(\frac{1}{(2\pi)^{D/2} |\boldsymbol{\Sigma}_k|^{1/2}} \right) - \frac{1}{2} (\dots)^\top \boldsymbol{\Sigma}_k^{-1} (\dots) \right]$$

$$= \sum_{n=1}^{N} q_{kn}^{(t)} \left[-\frac{1}{2} \log |\boldsymbol{\Sigma}_k| - \frac{1}{2} (\mathbf{x}_n - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k) \right]$$

This objective is a weighted log-likelihood for a Gaussian distribution.

- The Maximum Likelihood Estimate (MLE) for the covariance of a Gaussian is the sample covariance.
- For our *weighted* data (where $\mathbf{x}_n$ has weight $q_{kn}^{(t)}$), the MLE is the *weighted sample covariance*.

By taking the derivative w.r.t. $\boldsymbol{\Sigma}_k$ (using matrix calculus), setting $\boldsymbol{\mu}_k = \boldsymbol{\mu}_k^{(t+1)}$ (the new mean we just found), and setting the result to 0, we get:

$$\boldsymbol{\Sigma}_k^{(t+1)} = \frac{\sum_{n=1}^{N} q_{kn}^{(t)} (\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})(\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})^\top}{\sum_{n=1}^{N} q_{kn}^{(t)}}$$

M-step Derivation: Mixture π_k

Goal: Maximize w.r.t. π_k, subject to the constraint $\sum_{k=1}^{K} \pi_k = 1$.

- Objective terms for π: $\mathcal{L}(\pi) = \sum_{n=1}^{N} \sum_{k=1}^{K} q_{kn}^{(t)} \log \pi_k$
- Constraint: $\sum_{k=1}^{K} \pi_k - 1 = 0$

We use a **Lagrange multiplier** λ, as suggested: $\mathcal{J}(\pi, \lambda) = \mathcal{L}(\pi) - \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$.

Take the partial derivative w.r.t. a single π_k and set to 0:

$$\frac{\partial \mathcal{J}}{\partial \pi_k} = \sum_{n=1}^{N} \frac{q_{kn}^{(t)}}{\pi_k} - \lambda \overset{\text{set}}{=} 0 \Longrightarrow \sum_{n=1}^{N} q_{kn}^{(t)} = \lambda \pi_k$$

Let $N_k = \sum_{n=1}^{N} q_{kn}^{(t)}$, then $N_k = \lambda \pi_k$. Sum over all k to find λ:

$$\sum_{k=1}^{K} N_k = \sum_{k=1}^{K} \lambda \pi_k \Longrightarrow \sum_{k=1}^{K} \sum_{n=1}^{N} q_{kn}^{(t)} = \lambda \underbrace{\sum_{k=1}^{K} \pi_k}_{=1}, \qquad \sum_{n=1}^{N} \underbrace{\sum_{k=1}^{K} q_{kn}^{(t)}}_{=1} = \lambda \Longrightarrow N = \lambda .$$

Substitute $\lambda = N$ back into $N_k = \lambda \pi_k$: $N_k = N \pi_k \Longrightarrow \pi_k^{(t+1)} = \frac{N_k}{N} = \frac{1}{N} \sum_{n=1}^{N} q_{kn}^{(t)}$.

Summary of EM for GMM

Initialize $\boldsymbol{\mu}^{(1)}, \boldsymbol{\Sigma}^{(1)}, \boldsymbol{\pi}^{(1)}$ and iterate between the E and M step, until $\mathcal{L}(\boldsymbol{\theta})$ stabilizes.

Summary of EM for GMM

Initialize $\boldsymbol{\mu}^{(1)}, \boldsymbol{\Sigma}^{(1)}, \boldsymbol{\pi}^{(1)}$ and iterate between the E and M step, until $\mathcal{L}(\boldsymbol{\theta})$ stabilizes.

1. E-step: Compute assignments $q_{kn}^{(t)}$ (posterior probabilities):

$$q_{kn}^{(t)} := \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{k=1}^{K} \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})} \tag{14}$$

Summary of EM for GMM

Initialize $\boldsymbol{\mu}^{(1)}, \boldsymbol{\Sigma}^{(1)}, \boldsymbol{\pi}^{(1)}$ and iterate between the E and M step, until $\mathcal{L}(\boldsymbol{\theta})$ stabilizes.

1. E-step: Compute assignments $q_{kn}^{(t)}$ (posterior probabilities):

$$q_{kn}^{(t)} := \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{k=1}^K \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})} \tag{14}$$

2. Compute the marginal likelihood (cost).

$$\mathcal{L}(\boldsymbol{\theta}^{(t)}) = \sum_{n=1}^N \log \sum_{k=1}^K \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}) \tag{15}$$

Summary of EM for GMM

Initialize $\boldsymbol{\mu}^{(1)}, \boldsymbol{\Sigma}^{(1)}, \boldsymbol{\pi}^{(1)}$ and iterate between the E and M step, until $\mathcal{L}(\boldsymbol{\theta})$ stabilizes.

1. E-step: Compute assignments $q_{kn}^{(t)}$ (posterior probabilities):

$$q_{kn}^{(t)} := \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{k=1}^K \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})} \tag{14}$$

2. Compute the marginal likelihood (cost).

$$\mathcal{L}(\boldsymbol{\theta}^{(t)}) = \sum_{n=1}^N \log \sum_{k=1}^K \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}) \tag{15}$$

3. M-step: Update $\boldsymbol{\mu}_k^{(t+1)}, \boldsymbol{\Sigma}_k^{(t+1)}, \pi_k^{(t+1)}$.

$$\boldsymbol{\mu}_k^{(t+1)} := \frac{\sum_{n=1}^N q_{kn}^{(t)} \mathbf{x}_n}{\sum_{n=1}^N q_{kn}^{(t)}}, \qquad \boldsymbol{\Sigma}_k^{(t+1)} := \frac{\sum_{n=1}^N q_{kn}^{(t)} (\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})(\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})^\top}{\sum_{n=1}^N q_{kn}^{(t)}}, \tag{16}$$

$$\pi_k^{(t+1)} := \frac{1}{N} \sum_{n=1}^N q_{kn}^{(t)} \tag{17}$$

Summary of EM for GMM

Initialize $\boldsymbol{\mu}^{(1)}, \boldsymbol{\Sigma}^{(1)}, \boldsymbol{\pi}^{(1)}$ and iterate between the E and M step, until $\mathcal{L}(\boldsymbol{\theta})$ stabilizes.

1. E-step: Compute assignments $q_{kn}^{(t)}$ (posterior probabilities):

$$q_{kn}^{(t)} := \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{k=1}^K \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})} \tag{14}$$

2. Compute the marginal likelihood (cost).

$$\mathcal{L}(\boldsymbol{\theta}^{(t)}) = \sum_{n=1}^N \log \sum_{k=1}^K \pi_k^{(t)} \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}) \tag{15}$$

3. M-step: Update $\boldsymbol{\mu}_k^{(t+1)}, \boldsymbol{\Sigma}_k^{(t+1)}, \pi_k^{(t+1)}$.

$$\boldsymbol{\mu}_k^{(t+1)} := \frac{\sum_{n=1}^N q_{kn}^{(t)} \mathbf{x}_n}{\sum_{n=1}^N q_{kn}^{(t)}}, \qquad \boldsymbol{\Sigma}_k^{(t+1)} := \frac{\sum_{n=1}^N q_{kn}^{(t)} (\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})(\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})^\top}{\sum_{n=1}^N q_{kn}^{(t)}}, \tag{16}$$

$$\pi_k^{(t+1)} := \frac{1}{N} \sum_{n=1}^N q_{kn}^{(t)} \tag{17}$$

If we let the covariance be diagonal i.e. $\boldsymbol{\Sigma}_k := \sigma^2 \mathbf{I}$, EM algorithm is same as K-means as $\sigma^2 \to 0$.

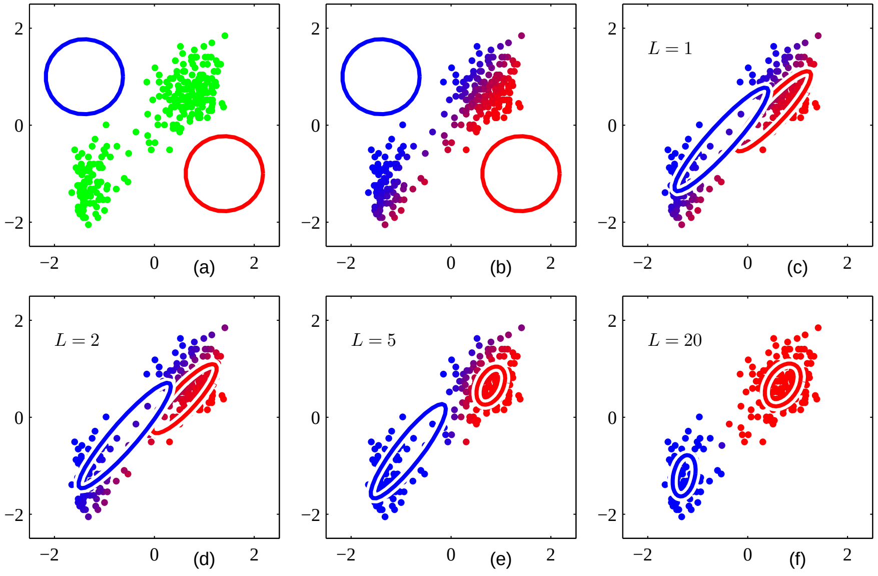
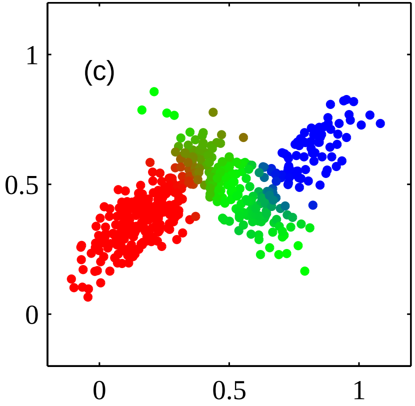


Figure: EM algorithm for GMM: Visualization of the iterative fitting process

Posterior distribution

We now show that $q_{kn}^{(t)}$ is the posterior distribution of the latent variable, i.e. $q_{kn}^{(t)} = p(z_n = k \mid \mathbf{x}_n, \boldsymbol{\theta}^{(t)})$

$$\underbrace{p(\mathbf{x}_n, z_n|\boldsymbol{\theta})}_{\text{joint}} = \underbrace{p(\mathbf{x}_n|z_n, \boldsymbol{\theta})}_{\text{likelihood}} \underbrace{p(z_n|\boldsymbol{\theta})}_{\text{prior}} = \underbrace{p(z_n|\mathbf{x}_n, \boldsymbol{\theta})}_{\text{posterior}} \underbrace{p(\mathbf{x}_n|\boldsymbol{\theta})}_{\text{marginal likelihood}} \tag{18}$$



The relationship to K-means

- The EM algorithm for GMMs is reminiscent of the K-means clustering algorithm.

The relationship to K-means

- The EM algorithm for GMMs is reminiscent of the K-means clustering algorithm.
- Instead of the "hard" cluster assignments in K-means, EM uses "soft" assignments q_{kn}.

$$q_{kn} = p(z_n = k \mid \mathbf{x}_n, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) \tag{19}$$

The relationship to K-means

- The EM algorithm for GMMs is reminiscent of the K-means clustering algorithm.
- Instead of the “hard” cluster assignments in K-means, EM uses “soft” assignments q_{kn}.

$$q_{kn} = p(z_n = k \mid \mathbf{x}_n, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) \tag{19}$$

- “Soft” means our assignments are probabilities taking values in $[0, 1]$.

The relationship to K-means

- The EM algorithm for GMMs is reminiscent of the K-means clustering algorithm.
- Instead of the “hard” cluster assignments in K-means, EM uses “soft” assignments q_{kn}.

$$q_{kn} = p(z_n = k \mid \mathbf{x}_n, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) \tag{19}$$

- “Soft” means our assignments are probabilities taking values in $[0, 1]$.
- “Hard” assignments take values in $\{0, 1\}$ or $\{1, \ldots, K\}$.

The relationship to K-means

- The EM algorithm for GMMs is reminiscent of the K-means clustering algorithm.
- Instead of the “hard” cluster assignments in K-means, EM uses “soft” assignments q_{kn}.

$$q_{kn} = p(z_n = k \mid \mathbf{x}_n, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) \tag{19}$$

- “Soft” means our assignments are probabilities taking values in $[0, 1]$.
- “Hard” assignments take values in $\{0, 1\}$ or $\{1, \ldots, K\}$.
- Like K-means, EM is also susceptible to local optima, so initializing at several different parameter values may be a good strategy.

EM Convergence Properties

- The EM algorithm is guaranteed to increase the likelihood at each iteration:

$$\mathcal{L}(\boldsymbol{\theta}^{(t+1)}) \geq \mathcal{L}(\boldsymbol{\theta}^{(t)}) \tag{20}$$

EM Convergence Properties

- The EM algorithm is guaranteed to increase the likelihood at each iteration:

$$\mathcal{L}(\boldsymbol{\theta}^{(t+1)}) \geq \mathcal{L}(\boldsymbol{\theta}^{(t)}) \tag{20}$$

- This guarantees convergence to a local maximum of the likelihood function.

EM Convergence Properties

- The EM algorithm is guaranteed to increase the likelihood at each iteration:

$$\mathcal{L}(\boldsymbol{\theta}^{(t+1)}) \geq \mathcal{L}(\boldsymbol{\theta}^{(t)}) \tag{20}$$

- This guarantees convergence to a local maximum of the likelihood function.
- However, like K-means, it may not find the global maximum.

EM Convergence Properties

- The EM algorithm is guaranteed to increase the likelihood at each iteration:

$$\mathcal{L}(\boldsymbol{\theta}^{(t+1)}) \geq \mathcal{L}(\boldsymbol{\theta}^{(t)}) \tag{20}$$

- This guarantees convergence to a local maximum of the likelihood function.
- However, like K-means, it may not find the global maximum.
- The rate of convergence depends on the fraction of information about the parameters contained in the latent variables.

EM Convergence Properties

- The EM algorithm is guaranteed to increase the likelihood at each iteration:

$$\mathcal{L}(\boldsymbol{\theta}^{(t+1)}) \geq \mathcal{L}(\boldsymbol{\theta}^{(t)}) \tag{20}$$

- This guarantees convergence to a local maximum of the likelihood function.
- However, like K-means, it may not find the global maximum.
- The rate of convergence depends on the fraction of information about the parameters contained in the latent variables.
- In practice, we monitor the change in log-likelihood and stop when it falls below a threshold.

EM and Variational Inference

- The ELBO perspective reveals that EM is a special case of variational inference.

EM and Variational Inference

- The ELBO perspective reveals that EM is a special case of variational inference.
- Variational inference approximates intractable posterior distributions with a tractable distribution $q(\mathbf{z})$.

EM and Variational Inference

- The ELBO perspective reveals that EM is a special case of variational inference.
- Variational inference approximates intractable posterior distributions with a tractable distribution $q(\mathbf{z})$.
- In standard EM:
  - We set $q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^{(t)})$ exactly in the E-step.
  - This is possible because the posterior is tractable for models like GMM.

EM and Variational Inference

- The ELBO perspective reveals that EM is a special case of variational inference.
- Variational inference approximates intractable posterior distributions with a tractable distribution $q(\mathbf{z})$.
- In standard EM:
  - We set $q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^{(t)})$ exactly in the E-step.
  - This is possible because the posterior is tractable for models like GMM.
- In variational EM:
  - The posterior $p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})$ is intractable.
  - We restrict $q(\mathbf{z})$ to a simpler family of distributions.
  - E-step: Find $q(\mathbf{z})$ that maximizes ELBO (minimizes KL-divergence).
  - M-step: Same as standard EM (maximize ELBO w.r.t. $\boldsymbol{\theta}$).

EM and Variational Inference

- The ELBO perspective reveals that EM is a special case of variational inference.
- Variational inference approximates intractable posterior distributions with a tractable distribution $q(\mathbf{z})$.
- In standard EM:
    - We set $q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^{(t)})$ exactly in the E-step.
    - This is possible because the posterior is tractable for models like GMM.
- In variational EM:
    - The posterior $p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})$ is intractable.
    - We restrict $q(\mathbf{z})$ to a simpler family of distributions.
    - E-step: Find $q(\mathbf{z})$ that maximizes ELBO (minimizes KL-divergence).
    - M-step: Same as standard EM (maximize ELBO w.r.t. $\boldsymbol{\theta}$).
- This connection has led to powerful techniques like variational autoencoders (VAEs).

Applications of EM algorithm

- The EM algorithm can be applied to many problems beyond GMMs:

Applications of EM algorithm

- The EM algorithm can be applied to many problems beyond GMMs:
  - Missing data problems

Applications of EM algorithm

- The EM algorithm can be applied to many problems beyond GMMs:
  - Missing data problems
  - Hidden Markov Models

Applications of EM algorithm

- The EM algorithm can be applied to many problems beyond GMMs:
  - Missing data problems
  - Hidden Markov Models
  - Mixture of experts models

Applications of EM algorithm

- The EM algorithm can be applied to many problems beyond GMMs:
    - Missing data problems
    - Hidden Markov Models
    - Mixture of experts models
    - Bayesian networks with hidden variables

Applications of EM algorithm

- The EM algorithm can be applied to many problems beyond GMMs:
  - Missing data problems
  - Hidden Markov Models
  - Mixture of experts models
  - Bayesian networks with hidden variables
- The same principles apply: identify latent variables, compute their expected values, and maximize parameters.

Practical Considerations for GMM Implementation

- **Initialization strategies:**

Practical Considerations for GMM Implementation

- **Initialization strategies:**
  - Random initialization of means and covariances

Practical Considerations for GMM Implementation

- **Initialization strategies:**
  - Random initialization of means and covariances
  - K-means initialization (use K-means clusters to initialize GMM parameters)

Practical Considerations for GMM Implementation

- **Initialization strategies:**
  - Random initialization of means and covariances
  - K-means initialization (use K-means clusters to initialize GMM parameters)
  - Multiple restarts to avoid local optima

Practical Considerations for GMM Implementation

- **Initialization strategies:**
  - Random initialization of means and covariances
  - K-means initialization (use K-means clusters to initialize GMM parameters)
  - Multiple restarts to avoid local optima
- **Numerical stability:**

Practical Considerations for GMM Implementation

- **Initialization strategies:**
  - Random initialization of means and covariances
  - K-means initialization (use K-means clusters to initialize GMM parameters)
  - Multiple restarts to avoid local optima
- **Numerical stability:**
  - Adding small values to diagonal of covariance matrices

Practical Considerations for GMM Implementation

- **Initialization strategies:**
  - Random initialization of means and covariances
  - K-means initialization (use K-means clusters to initialize GMM parameters)
  - Multiple restarts to avoid local optima
- **Numerical stability:**
  - Adding small values to diagonal of covariance matrices
  - Performing computations in log space to avoid underflow

Practical Considerations for GMM Implementation

- **Initialization strategies:**
  - Random initialization of means and covariances
  - K-means initialization (use K-means clusters to initialize GMM parameters)
  - Multiple restarts to avoid local optima
- **Numerical stability:**
  - Adding small values to diagonal of covariance matrices
  - Performing computations in log space to avoid underflow
- **Model selection:**

Practical Considerations for GMM Implementation

- **Initialization strategies:**
  - Random initialization of means and covariances
  - K-means initialization (use K-means clusters to initialize GMM parameters)
  - Multiple restarts to avoid local optima
- **Numerical stability:**
  - Adding small values to diagonal of covariance matrices
  - Performing computations in log space to avoid underflow
- **Model selection:**
  - Determining the optimal number of components K

Practical Considerations for GMM Implementation

- **Initialization strategies:**
  - Random initialization of means and covariances
  - K-means initialization (use K-means clusters to initialize GMM parameters)
  - Multiple restarts to avoid local optima
- **Numerical stability:**
  - Adding small values to diagonal of covariance matrices
  - Performing computations in log space to avoid underflow
- **Model selection:**
  - Determining the optimal number of components K
  - Using criteria like BIC (Bayesian Information Criterion) or AIC (Akaike Information Criterion)

Table of Contents

A Deeper Look: Unifying K-means and GMM

At first glance, K-means and GMM (via EM) seem very different:

K-means

- **Goal:** Minimize Sum of Squares
- **Loss:** $\mathcal{L} = \sum_{n,k} z_{nk} \left\|\mathbf{x}_n - \boldsymbol{\mu}_k\right\|^2$
- **Method:** Simple geometry, hard assignments.

GMM (EM)

- **Goal:** Maximize Log-Likelihood
- **Loss:** $\mathcal{L} = \sum_n \log \sum_k \pi_k \mathcal{N}(\dots)$
- **Method:** Complex, probabilistic, soft assignments.

A Deeper Look: Unifying K-means and GMM

At first glance, K-means and GMM (via EM) seem very different:

K-means

- **Goal:** Minimize Sum of Squares
- **Loss:** $\mathcal{L} = \sum_{n,k} z_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$
- **Method:** Simple geometry, hard assignments.

GMM (EM)

- **Goal:** Maximize Log-Likelihood
- **Loss:** $\mathcal{L} = \sum_n \log \sum_k \pi_k \mathcal{N}(\dots)$
- **Method:** Complex, probabilistic, soft assignments.

Key Idea: What if they are both instances of the *same* general algorithm, just using different “distance” functions?

The Unifying Tool: Bregman Divergence

A Bregman divergence is a “distance-like” function (not a true distance, as $D(p,q) \neq D(q,p)$) generated by any strictly convex function F.

Definition:

$$D_F(p,q) = \underbrace{F(p)}_{\text{Function at p}} - \Big[\underbrace{F(q) + \nabla F(q)^\top (p-q)}_{\text{Taylor approx. at } q\text{, eval. at } p}\Big]$$

Geometric Intuition: It is the vertical gap between the function F at point p and the tangent line of F at point q.

Connection 1: K-means is Bregman Clustering

K-means minimizes the Squared Euclidean Distance.

Claim: The Squared Euclidean Distance *is* a Bregman divergence.
Proof:

- Let the convex function be $F(\mathbf{x}) = \|\mathbf{x}\|^2 = \mathbf{x}^\top \mathbf{x}$.
- Its gradient is $\nabla F(\mathbf{x}) = 2\mathbf{x}$.

Now, plug this into the definition of $D_F(\mathbf{x}, \boldsymbol{\mu})$:

$$\begin{aligned} D_F(\mathbf{x}, \boldsymbol{\mu}) &= F(\mathbf{x}) - \left[F(\boldsymbol{\mu}) + \nabla F(\boldsymbol{\mu})^\top (\mathbf{x} - \boldsymbol{\mu})\right] \\ &= \|\mathbf{x}\|^2 - \left[\|\boldsymbol{\mu}\|^2 + (2\boldsymbol{\mu})^\top (\mathbf{x} - \boldsymbol{\mu})\right] = \|\mathbf{x}\|^2 - \|\boldsymbol{\mu}\|^2 - 2\boldsymbol{\mu}^\top \mathbf{x} + 2\|\boldsymbol{\mu}\|^2 \\ &= \|\mathbf{x}\|^2 - 2\boldsymbol{\mu}^\top \mathbf{x} + \|\boldsymbol{\mu}\|^2 = \|\mathbf{x} - \boldsymbol{\mu}\|_2^2 \end{aligned}$$

K-means is a clustering algorithm that
minimizes the Bregman divergence generated by $F(\mathbf{x}) = \|\mathbf{x}\|^2$.

Connection 2: GMM (EM) is Bregman Clustering

GMM-EM maximizes the log-likelihood, which is equivalent to minimizing the negative log-likelihood.

Key Insight:

- The **negative log-likelihood** of *any* distribution in the exponential family is a Bregman divergence.
- A Gaussian distribution $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is in the exponential family.
- Therefore, the GMM "distance" (loss) for a point $\mathbf{x}_n$ to a cluster k is also a Bregman divergence:

$$D_{\text{GMM}}(\mathbf{x}_n, \theta_k) = -\log \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Connecting to K-means:

- If we set $\boldsymbol{\Sigma}_k = \sigma^2\mathbf{I}$ (spherical clusters) and $\sigma^2 \to 0$, this divergence becomes:

$$D_{\text{GMM}} \propto \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2$$

- This exactly matches previous slide's conclusion: GMM-EM becomes K-means when $\boldsymbol{\Sigma}_k$ is isotropic and $\sigma^2 \to 0$.

The Unification: Hard vs. Soft Clustering

Both algorithms are just “Bregman clustering”, differing only in the *divergence* used and the *assignment* type.

K-means (Hard Clustering)

- **Divergence:** $D_F(\mathbf{x}, \boldsymbol{\mu}) = \|\mathbf{x} - \boldsymbol{\mu}\|^2$
- **E-step:** Hard Assignment. Find the *single* k that minimizes the divergence:

$$z_{nk} = 1 \Longleftrightarrow k = \arg\min_j D_F(\mathbf{x}_n, \boldsymbol{\mu}_j)$$

- **M-step:** Update $\boldsymbol{\mu}_k$ to be the mean (centroid) of all points assigned to it.

GMM (EM) (Soft Clustering)

- **Divergence:** $D_{\text{GMM}}(\mathbf{x}, \theta_k) = -\log \mathcal{N}(\mathbf{x}|\theta_k)$
- **E-step:** Soft Assignment. Compute $q_{kn}^{(t)}$ (responsibility) based on the divergence (and prior π_k).

$$q_{kn}^{(t)} \propto \pi_k \cdot \exp(-D_{\text{GMM}}(\mathbf{x}_n, \theta_k))$$

- **M-step:** Update θ_k to be the *weighted* mean (centroid) of all points.