Mixture Models and EM

(Part 1)

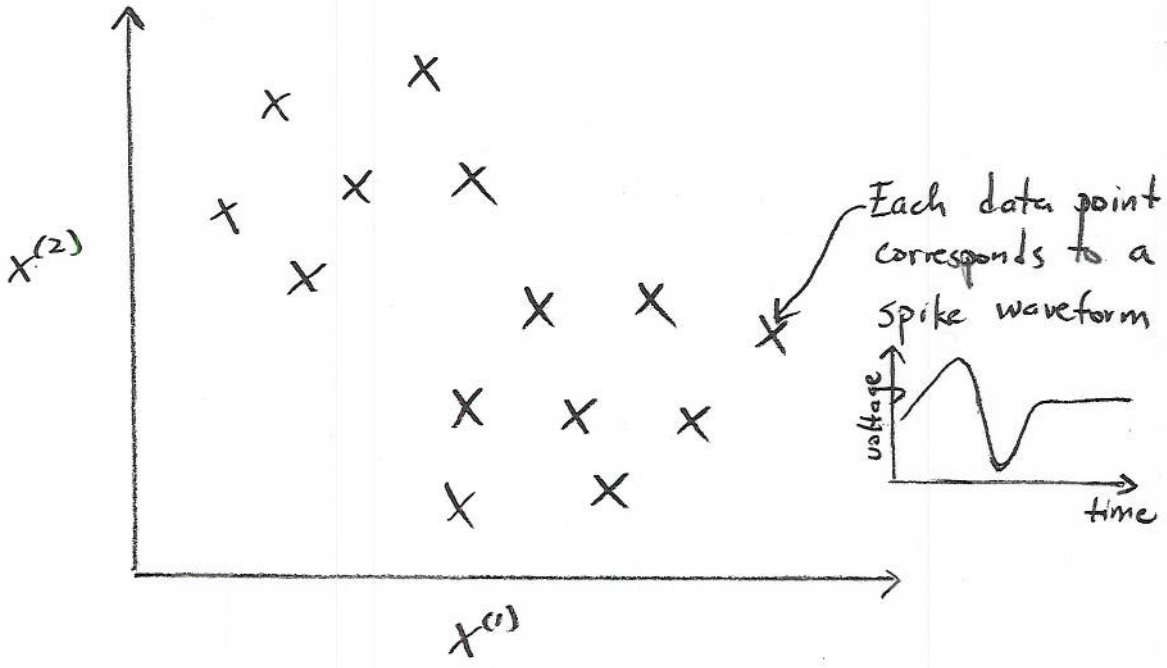
A) K-means Clustering

Suppose we have a data set $\underline{x}_n \in \mathbb{R}^D$ where $n = 1, \ldots, N$.

Goal: Partition the data set into some number K of clusters.

For now, assume K is given.

Picture to have in mind:



How would you partition this dataset into $K = 2$ clusters?

Intuitive definition of a cluster:

A group of data points whose inter-point distances are small compared with the distances to points outside the cluster.

Let's formalize this notion:

Let $\underline{\mu}_k \in \mathbb{R}^D$ where $k = 1, \ldots, K$ be the "prototype" associated with the k^{th} cluster.

$$r_{nk} = \begin{cases} 1 & \text{if } \underline{x}_n \text{ belongs to the } k^{th} \text{ cluster} \\ 0 & \text{else} \end{cases}$$

objective function →

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \| \underline{x}_n - \underline{\mu}_k \|^2$$

Find $\{r_{nk}\}$ and $\{\underline{\mu}_k\}$ such that J is minimized.

This can be solved by iterating:

- Minimize J w.r.t. r_{nk}, keeping $\underline{\mu}_k$ fixed ("E-step")
- Minimize J w.r.t. $\underline{\mu}_k$, keeping r_{nk} fixed ("M-step")

A.1) E-step for K-means

Constraint: $\{r_{n1}, \ldots, r_{nK}\}$ is a set of $(K-1)$ zeros and a single 1.

Can optimize J wrt. r_{nk} for each n separately.

For each n, assign:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \underset{j}{\operatorname{argmin}} \|\underline{x}_n - \underline{\mu}_j\|^2 \\ 0 & \text{else} \end{cases}$$

In words: Assign each data point to the closest cluster center.

A.2) M-step for K-means

$$\frac{\partial J}{\partial \underline{\mu}_k} = 2 \sum_{n=1}^{N} r_{nk} (\underline{x}_n - \underline{\mu}_k) = 0$$

$$\boxed{\underline{\mu}_k = \frac{\sum_{n=1}^{N} r_{nk} \underline{x}_n}{\sum_{n=1}^{N} r_{nk}}}$$

In words: Set $\underline{\mu}_k$ equal to the mean of all data points assigned to cluster k.

A.3) Convergence of the K-means algorithm.

- The E-step and M-step should be iterated until there is no further change in cluster assignments, or until some maximum number of iterations is exceeded.

- Each iteration reduces the objective function J, so the algorithm is <u>guaranteed to converge</u>. (We will prove this later in the context of the EM algorithm.)

- Convergence is guaranteed to a <u>local</u> (rather than a global) minimum of J.

- If there are multiple local optima, the particular local optimum reached depends on the <u>initialization</u> of the $\{\underline{\mu}_k\}$.

[Let's look at an example of K-means in action. PRML Figures 9.1 and 9.2.]

Mixture Models and EM

(Part 2)

B) Mixtures of Gaussians

Let $z \in \{1, \ldots, K\}$ be a random variable

$$P(z=k) = \pi_k \quad , \text{ where } k = 1, \ldots, K$$

(Note: For $P(z)$ to be a valid probability distribution:

$$0 \leq \pi_k \leq 1$$

$$\sum_{k=1}^{K} \pi_k = 1$$

)

$$P(\underline{x} \mid z=k) = N(\underline{x} \mid \underline{\mu}_k, \Sigma_k)$$

Thus,

$$P(\underline{x}) = \sum_z P(\underline{x} \mid z) P(z) = \sum_{k=1}^{K} N(\underline{x} \mid \underline{\mu}_k, \Sigma_k) \cdot \pi_k$$

What is the graphical model for $P(\underline{x}, z)$?

B.1) Maximum likelihood parameter estimation

Goal: Fit μ_k, Σ_k, π_k to training data $\underline{x}_1, \ldots, \underline{x}_N$.

$$P(\{\underline{x}\} \mid \theta) = \prod_{n=1}^{N} P(\underline{x}_n)$$

($\{\underline{x}\}$: shorthand for $\underline{x}_1, \ldots, \underline{x}_N$; θ: shorthand for $\mu_k, \Sigma_k, \pi_k \ (k=1, \ldots, K)$)

$$= \prod_{n=1}^{N} \sum_{k=1}^{K} N(\underline{x}_n \mid \mu_k, \Sigma_k) \cdot \pi_k$$

$$\underbrace{\log P(\{\underline{x}\} \mid \theta)}_{\text{call this } \mathcal{L}} = \sum_{n=1}^{N} \log \left\{ \sum_{k=1}^{K} N(\underline{x}_n \mid \mu_k, \Sigma_k) \cdot \pi_k \right\}$$

i) Find μ_k

$$\frac{\partial \mathcal{L}}{\partial \mu_k} = \sum_{n=1}^{N} \frac{1}{\sum_{j=1}^{K} N(\underline{x}_n \mid \mu_j, \Sigma_j) \cdot \pi_j} \cdot \pi_k \cdot \left(\frac{\partial}{\partial \mu_k} N(\underline{x}_n \mid \mu_k, \Sigma_k) \right) \qquad (1)$$

$$\frac{\partial}{\partial \underline{\mu}_k} N(\underline{x}_n | \underline{\mu}_k, \Sigma_k) = \frac{\partial}{\partial \underline{\mu}_k} \left(\frac{1}{(2\pi)^{\frac{D}{2}} |\Sigma_k|^{\frac{1}{2}}} e^{-\frac{1}{2}(\underline{x}_n - \underline{\mu}_k)^T \Sigma_k^{-1} (\underline{x}_n - \underline{\mu}_k)} \right)$$

$$= N(\underline{x}_n | \underline{\mu}_k, \Sigma_k) \cdot \Sigma_k^{-1} (\underline{x}_n - \underline{\mu}_k)$$

Plugging into (1),

$$\frac{\partial \mathcal{L}}{\partial \underline{\mu}_k} = \sum_{n=1}^{N} \underbrace{\frac{N(\underline{x}_n | \underline{\mu}_k, \Sigma_k) \cdot \pi_k}{\sum_{j=1}^{K} N(\underline{x}_n | \underline{\mu}_j, \Sigma_j) \cdot \pi_j}}_{\text{call this } \gamma_{nk}} \cdot \Sigma_k^{-1} (\underline{x}_n - \underline{\mu}_k)$$

$$= \Sigma_k^{-1} \sum_{n=1}^{N} \gamma_{nk} (\underline{x}_n - \underline{\mu}_k)$$

$$= 0$$

Letting $N_k = \sum_{n=1}^{N} \gamma_{nk}$,

$$\boxed{\underline{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} \underline{x}_n} \qquad (2)$$

Let's try to understand this result intuitively.

$$\gamma_{nk} = P(z_n = k \mid \underline{x}_n)$$

$$= \frac{P(\underline{x}_n \mid z_n = k)\, P(z_n = k)}{P(\underline{x}_n)}$$

$$= \frac{N(\underline{x}_n \mid \underline{\mu}_k, \Sigma_k) \cdot \pi_k}{\sum_{j=1}^{K} N(\underline{x}_n \mid \underline{\mu}_j, \Sigma_j) \cdot \pi_j} \qquad \left(\begin{array}{c} \underline{\text{Note:}} \\ 0 \le \gamma_{nk} \le 1 \\ \sum_{k=1}^{K} \gamma_{nk} = 1 \end{array} \right)$$

γ_{nk} is the "responsibility" that cluster k takes for explaining the observation $\underline{x}_n$.

Whereas π_k is the <u>prior</u> probability that $z_n = k$, γ_{nk} is the <u>posterior</u> probability that $z_n = k$ once we have observed $\underline{x}_n$.

N_k is the effective number of data points assigned to cluster k.

Thus, in (2), $\underline{\mu}_k$ is obtained by taking a weighted mean, where the weights are given by the responsibilities ($0 \le \gamma_{nk} \le 1$).
(0: not responsible; 1: very responsible)

This is reminiscent of k-means, but weights can now range from 0 to 1 (soft assignments).

ii) Find Σ_k

Setting $\frac{\partial \mathcal{L}}{\partial \Sigma_k} = 0$,

$$\boxed{\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} (\underline{x}_n - \underline{\mu}_k)(\underline{x}_n - \underline{\mu}_k)^T} \quad (3)$$

(we won't show this here)

(3) has an intuitive interpretation like in (2).

iii) Find π_k

Here, we need to perform a constrained optimization since $\sum_{k=1}^{K} \pi_k = 1$.

Instead of maximizing $\mathcal{L}$, we will maximize

$$\mathcal{L}' = \mathcal{L} + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right)$$

λ: Lagrange multiplier

$$\frac{\partial \mathcal{L}'}{\partial \pi_k} = \frac{\partial \mathcal{L}}{\partial \pi_k} + \lambda$$

$$= \sum_{n=1}^{N} \frac{1}{\sum_{j=1}^{N} N(\underline{x}_n | \underline{\mu}_j, \Sigma_j)\pi_j} \cdot N(\underline{x}_n | \underline{\mu}_k, \Sigma_k) + \lambda$$

$$= 0$$

Multiplying both sides by π_k,

$$\sum_{n=1}^{N} \gamma_{nk} + \lambda \cdot \pi_k = 0$$

$$\pi_k = -\frac{N_k}{\lambda} \qquad (4)$$

Enforcing the constraint $\sum_{k=1}^{K} \pi_k = 1$,

$$-\frac{\sum_{k=1}^{K} N_k}{\lambda} = 1$$

$$\lambda = -N$$

Plugging into (4),

$$\boxed{\pi_k = \frac{N_k}{N}} \qquad (5)$$

This is the effective number of data points assigned to class k, divided by the total number of data points.

Very important note:

(2), (3), and (5) do not constitute a closed-form solution because the responsibilities γ_{nk} (which appear in (2), (3), and (5)) depend on $\underline{\mu}_k$, Σ_k, and π_k.

This suggests an iterative scheme, which turns out to be an instance of the Expectation-Maximization (EM) algorithm.

The EM algorithm is a powerful and general method for finding the maximum likelihood parameters for models with latent variables.

In the mixture of Gaussians, $\underline{z}$ is a latent variable because it is not observed (only $\underline{x}$ is observed).

B.2) EM algorithm for Mixture of Gaussians

1) Initialize $\underline{\mu}_k, \Sigma_k, \pi_k$.

2) **E-step**: Evaluate responsibilities given current parameter values

$$\gamma_{nk} = \frac{N(\underline{x}_n | \underline{\mu}_k, \Sigma_k)\pi_k}{\sum_{j=1}^{K} N(\underline{x}_n | \underline{\mu}_j, \Sigma_j)\pi_j}$$

3) **M-step**: Re-estimate parameters using current responsibilities

$$\underline{\mu}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} \cdot \underline{x}_n$$

$$\Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} (\underline{x}_n - \underline{\mu}_k^{new})(\underline{x}_n - \underline{\mu}_k^{new})^T$$

$$\pi_k^{new} = \frac{N_k}{N}$$

where $N_k = \sum_{n=1}^{N} \gamma_{nk}$

4) Evaluate log likelihood of data

$$\log P(\{\underline{x}\}|\theta) = \sum_{n=1}^{N} \log \left\{ \sum_{k=1}^{K} N(\underline{x}_n | \underline{\mu}_k, \Sigma_k) \cdot \pi_k \right\}$$

and check for convergence of either the parameters or the log likelihood. If convergence criterion not satisfied, return to step 2).

- We will soon show that each iteration of the EM algorithm is guaranteed to increase the log likelihood $\log P(\{\underline{x}\}|\theta)$.

- See PRML Figure 9.8 for an example of EM in action as applied to a mixture of Gaussians.

Mixture Models and EM

Neural Signal Processing
Prof. Byron Yu

(Part 3)

C) Relating EM for Gaussian Mixtures to Classification ← call this ①

Call this ②

During the training phase, the goal in both cases is to estimate the model parameters from the training data.

Key difference: In ①, the class labels are known.
In ②, the class labels are not known.

In ①, we were able to estimate the model parameters in closed form:

$$\pi_k = \frac{N_k}{N}, \text{ where } N_k = \sum_{n \in C_k} 1 \quad (1)$$

$$\underline{\mu}_k = \frac{1}{N_k} \sum_{n \in C_k} \underline{x}_n \quad (2)$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n \in C_k} (\underline{x}_n - \underline{\mu}_k)(\underline{x}_n - \underline{\mu}_k)^T \quad (3)$$

In ②, it is unknown whether $n \in C_k$ (i.e. we don't have class labels), which leads to a "chicken and egg" problem.

- If the class labels were known, it would be easy to compute model parameters using (1)-(3).
- If the model parameters were known, it would be easy to compute a distribution on the class labels $P(C_k \mid \underline{x})$. (see p.9 of Classification notes)

Using the EM algorithm, we can "bootstrap" our way out of the chicken and egg problem.

1) Initialize model parameters

E-step 2) Estimate class labels given current model parameters.

M-step 3) Estimate model parameters given current class labels (represented as a distribution over class labels).

4) Go to 2).

In ②, the <u>training phase</u> uses the EM algorithm, in which:

- The E-step is reminiscent of test phase in ①
- The M-step is reminiscent of training phase in ①.

In ②, the <u>test phase</u> typically involves running a single E-step on the test data (no iterating needed).

D) The EM Algorithm in General

D.1) Motivation

You have a model that you would like to fit to your training data.

If the model has latent variables (<u>very</u> likely in neural signal processing), the EM algorithm provides a recipe for fitting the model.

Because of its generality, the EM algorithm is among the most important and widely used tools in machine learning.

We will be using it repeatedly in the rest of this course.

D2) Decomposition of data likelihood

Let X denote all observed variables

Z denote all latent variables

θ denote all model parameters.

<u>Goal</u>: Maximize $p(X|\theta)$ w.r.t. θ.

Let $q(Z)$ be some distribution over latent variables.

For any choice of $q(Z)$, we can decompose $\log p(X|\theta)$ into a sum of two terms:

$$\log p(X|\theta) = \mathcal{L}(q, \theta) + KL(q \| p), \quad (4)$$

where

"Joint distribution"

$$\mathcal{L}(q, \theta) = \sum_{z} q(z) \log \frac{p(X, z \mid \theta)}{q(z)} \qquad (5)$$

"Posterior distribution"

$$KL(q \| p) = -\sum_{z} q(z) \log \frac{p(z \mid X, \theta)}{q(z)} \qquad (6)$$

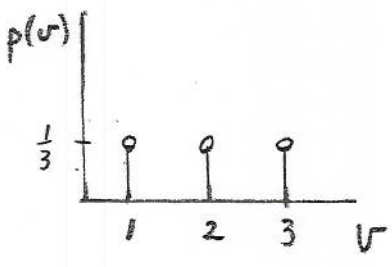
Notes:

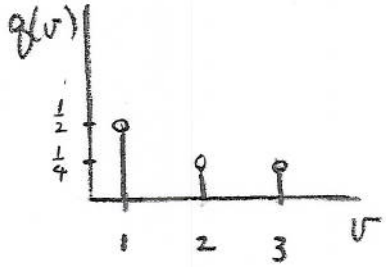
- $\mathcal{L}(q, \theta)$ is a scalar that depends only on q and θ because X is observed and z is summed out.
- $KL(q \| p)$ is the Kullback-Leibler divergence between $q(z)$ and $p(z \mid X, \theta)$.

The KL divergence is a scalar that measures the distance between probability distributions. In general, KL divergence is written:

$$KL(q \| p) = -\sum_{v} q(v) \log \frac{p(v)}{q(v)}$$

Let's do a simple example to illustrate KL divergence.





$$KL(q\|p) = -\left(q(1)\log\frac{p(1)}{q(1)} + q(2)\log\frac{p(2)}{q(2)} + q(3)\log\frac{p(3)}{q(3)} \right)$$

$$= -\left(\frac{1}{2}\cdot\log\frac{\frac{1}{3}}{\frac{1}{2}} + \frac{1}{4}\log\frac{\frac{1}{3}}{\frac{1}{4}} + \frac{1}{4}\log\frac{\frac{1}{3}}{\frac{1}{4}} \right)$$

$$= 0.0589$$

$$KL(p\|p) = -\sum_{v} p(v)\log\frac{p(v)}{p(v)} = 0$$

($\frac{p(v)}{p(v)} \to 1$, $\log 1 \to 0$)

<u>Facts</u>: $KL(q\|p) \geq 0$ for any p and q.

$KL(q\|p) = 0$ iff $p = q$.

- In (6), $KL(q\|p) \geq 0$, with equality iff $q(z) = p(z|x,\theta)$.
- Thus, $\log p(x|\theta) \geq \mathcal{L}(q,\theta)$

  $\Rightarrow$ $\mathcal{L}(q,\theta)$ is a lower bound for $\log p(x|\theta)$
- The lower bound is tight (i.e. $\log p(x|\theta) = \mathcal{L}(q,\theta)$) when $q(z)$ is chosen to be $p(z|x,\theta)$.

Let's verify the decomposition (4):

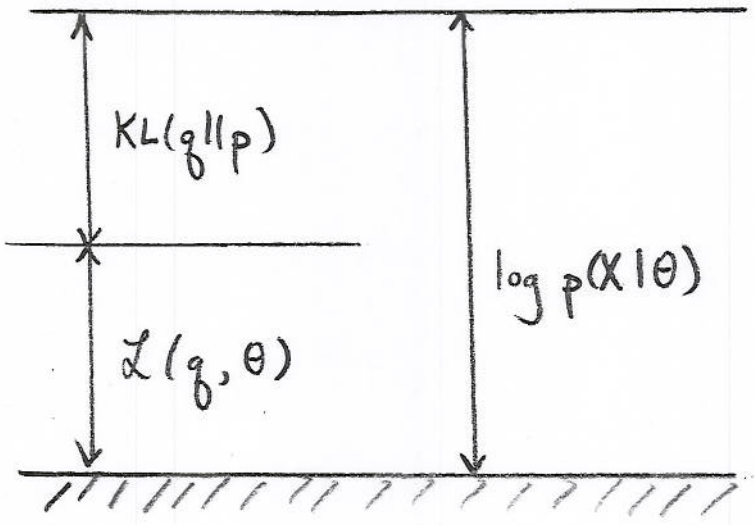
$$\mathcal{L}(q,\theta) = \sum_z q(z) \log \frac{p(z|x,\theta)\, p(x|\theta)}{q(z)}$$

$$= \sum_z q(z) \log \frac{p(z|x,\theta)}{q(z)} + \sum_z q(z) \log p(x|\theta)$$

$$= -KL(q\|p) + \log p(x|\theta) \cdot \underbrace{\sum_z q(z)}_{\to 1}$$

$$\Rightarrow \log p(x|\theta) = \mathcal{L}(q,\theta) + KL(q\|p). \;//$$

Picture to have in mind



D.3) The EM algorithm

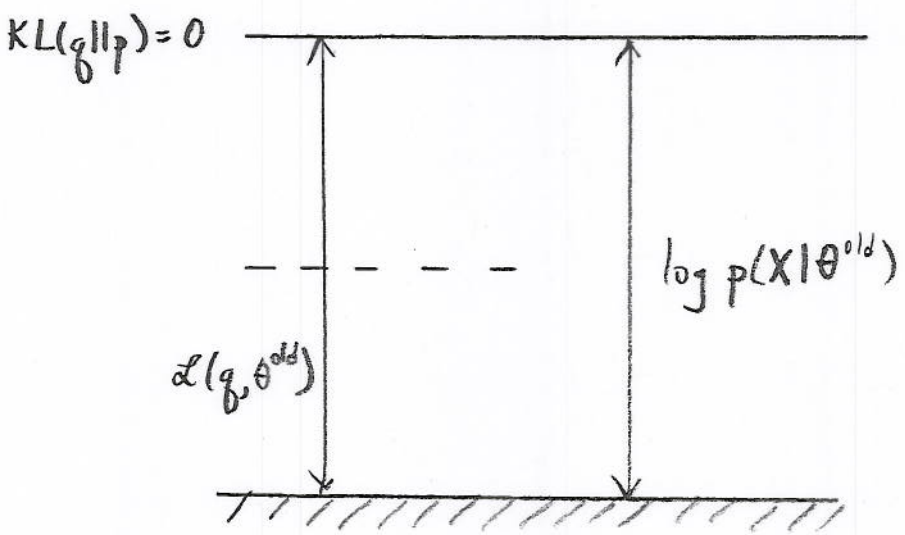
Goal: Maximize $\log p(X|\theta)$ w.r.t. θ

Approach: Iteratively maximize the lower bound $\mathcal{L}(q, \theta)$ w.r.t. q and θ.

- E-step: maximize $\mathcal{L}(q, \theta)$ w.r.t. q while keeping θ fixed.
- M-step: maximize $\mathcal{L}(q, \theta)$ w.r.t. θ while keeping q fixed.

E-step

With fixed θ, $\mathcal{L}(q, \theta)$ is maximized when $KL(q||p) = 0$, corresponding to $q(Z) = p(Z|X, \theta)$.

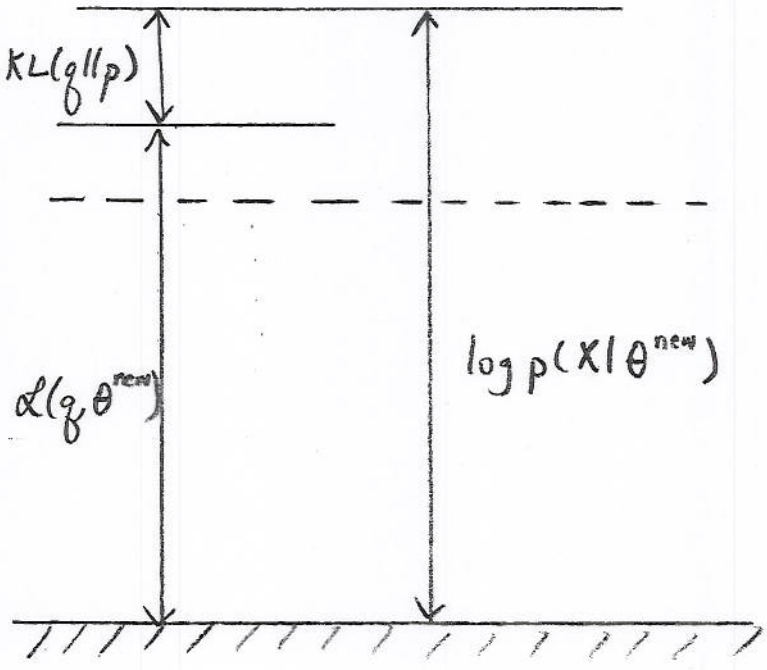


M-step

$$\mathcal{L}(q,\theta) = \sum_Z q(Z) \log p(X,Z|\theta) - \underbrace{\sum_Z q(Z) \log q(Z)}_{\text{no } \theta \text{ dependence}}$$

With fixed q, maximizing $\mathcal{L}(q,\theta)$ is equivalent to maximizing $\underbrace{\sum_Z q(Z) \log p(X,Z|\theta)}$.

This is $E_q[\log p(X,Z|\theta)]$, the "expected log joint distribution".

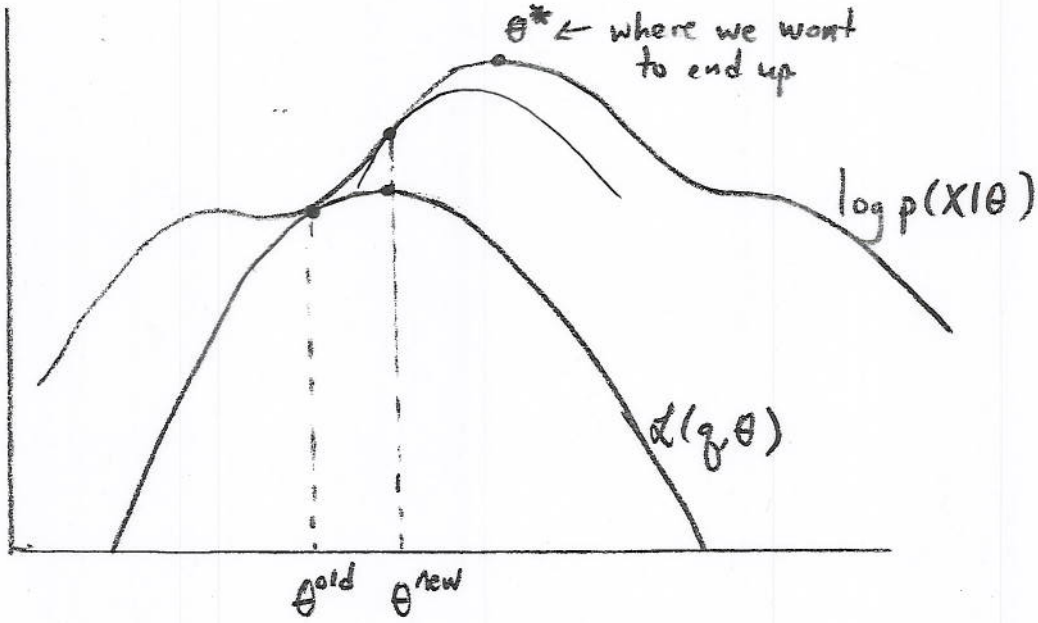


- Because $\mathcal{L}(q,\theta)$ is lower bound on $\log p(X|\theta)$, $\log p(X|\theta)$ must increase at least as much as $\mathcal{L}(q,\theta)$ does.
- Now, there's a nonzero $KL(q||p)$.

Based on the pictures on the previous 2 pages, we see that $\log p(X|\theta)$ is guaranteed to be non-decreasing from one EM iteration to the next.

Thus, the EM algorithm is guaranteed to converge to a local optimum.

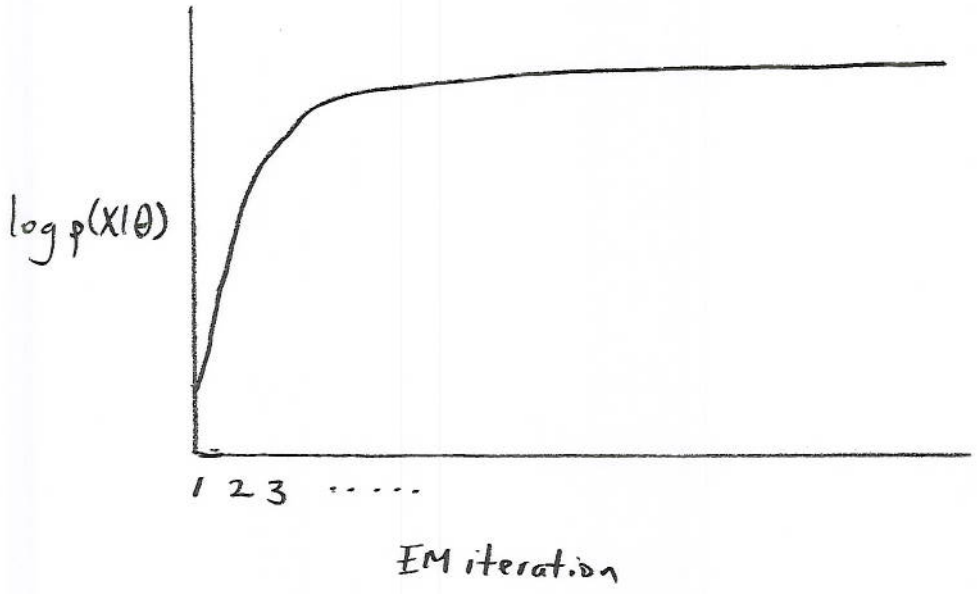
The EM algorithm in parameter space:



E-step: Make lower bound tight

M-step: Find peak of lower bound

Tracking the progress of the EM algorithm:



- This curve is guaranteed to be non-decreasing
- This is referred to as the "learning curve".

D.4) Summary of the general EM algorithm

1) Initialize parameters θ^{old}, Compute $\log p(X|\theta^{old})$.

2) E-step:

Evaluate $p(Z|X, \theta^{old})$

3) M-step:

Find $\theta^{new} = \arg\max_{\theta} \left(\sum_{Z} p(Z|X, \theta^{old}) \log p(X, Z|\theta) \right)$

4) Compute $\log p(X|\theta^{new})$. Compare to $\log p(X|\theta^{old})$.

5) If not converged, assign

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

and go to 2).