Introduction to Deep Generative Modeling

Lecture #4

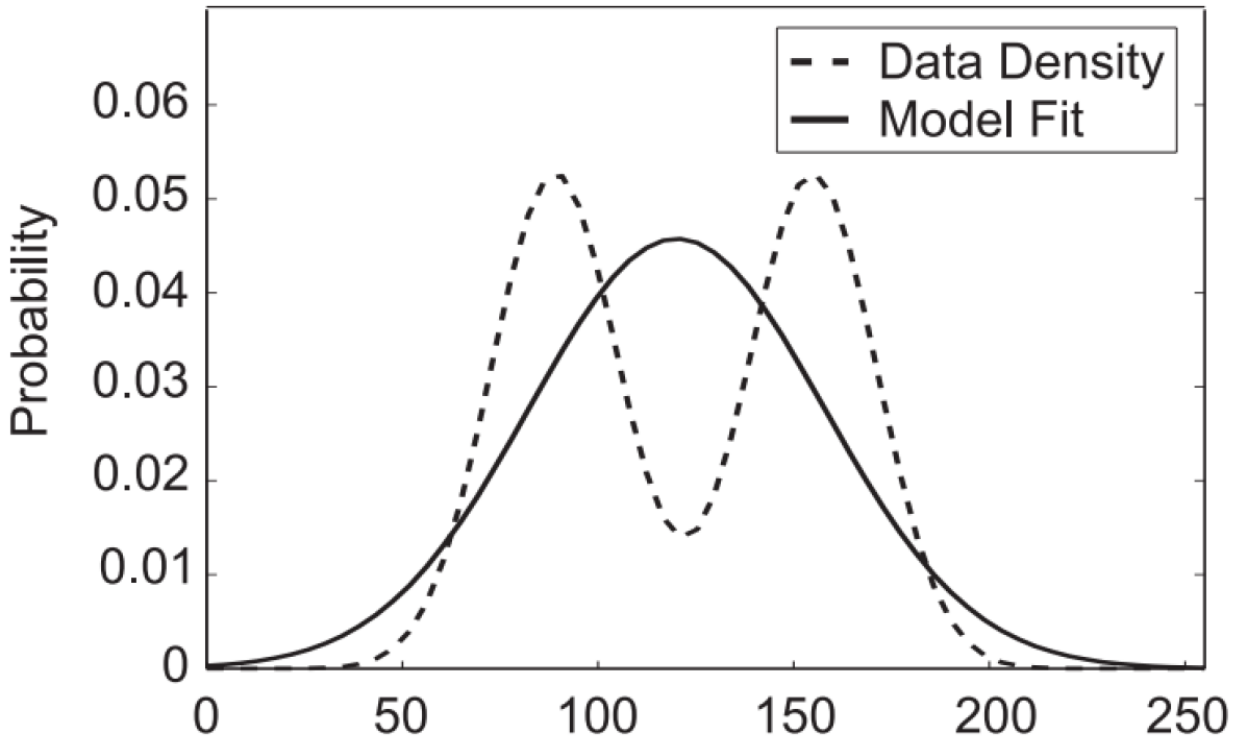
HY-673 – Computer Science Dep., University of Crete

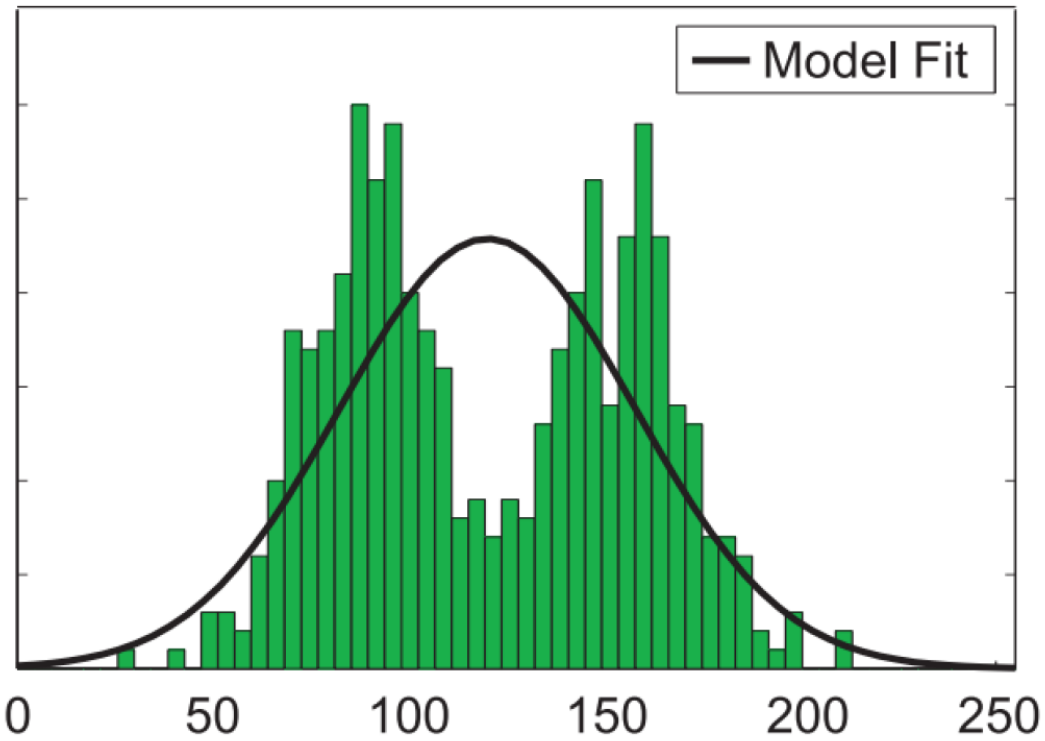
Professors: Yannis Pantazis & Yannis Stylianou

TAs: Michail Raptakis & Michail Spanakis

Gaussian Mixture Model - Motivation

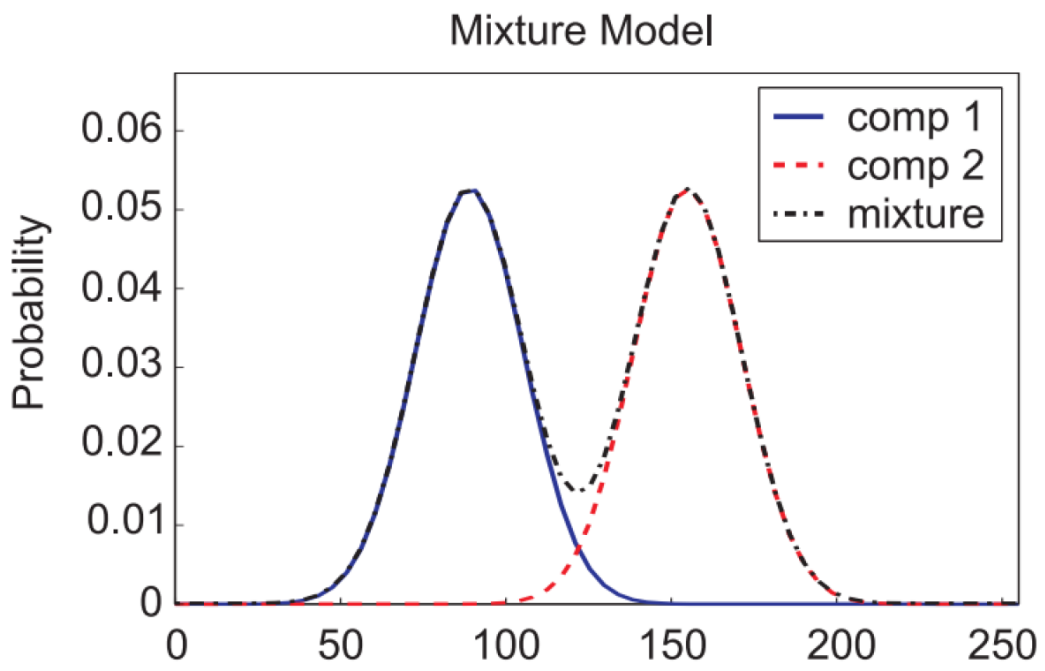
- Fitting a Gaussian to multimodal data is... a bad idea:

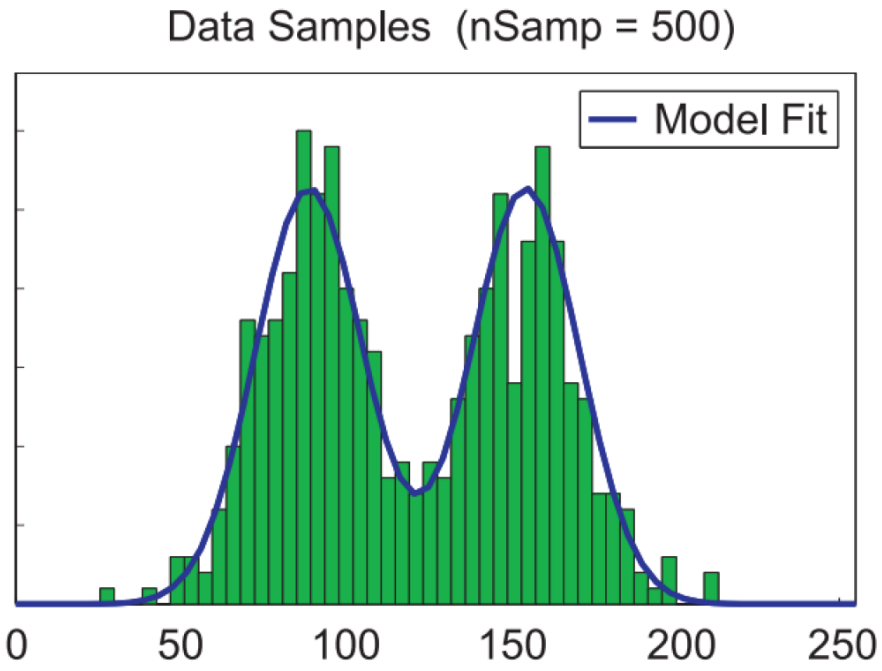




Gaussian Mixture Model - Motivation

- Now, we are trying to fit a GMM with $K = 2$ components (or modes):





GMM Definition

- The probability density function of a **Gaussian Mixture Model** (GMM) with K components is given by:

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

with π_k being the **mixing coefficients** or **weights** which satisfy

$$\sum_{k=1}^{K} \pi_k = 1 \text{ and } \pi_k \geq 0 \ \forall k.$$

while θ collects all the parameters of the model:

$$\theta := \{\pi_k, \mu_k, \Sigma_k, k = 1, ..., K\}.$$

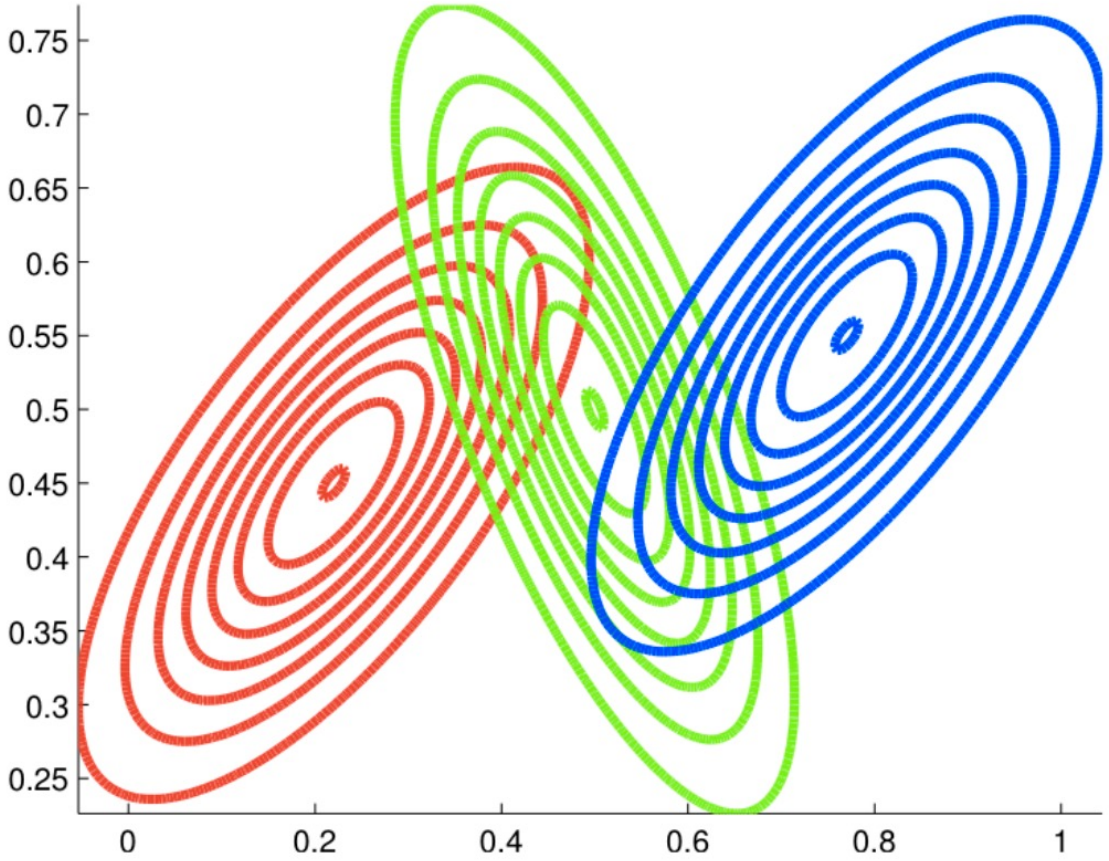
GMM Properties

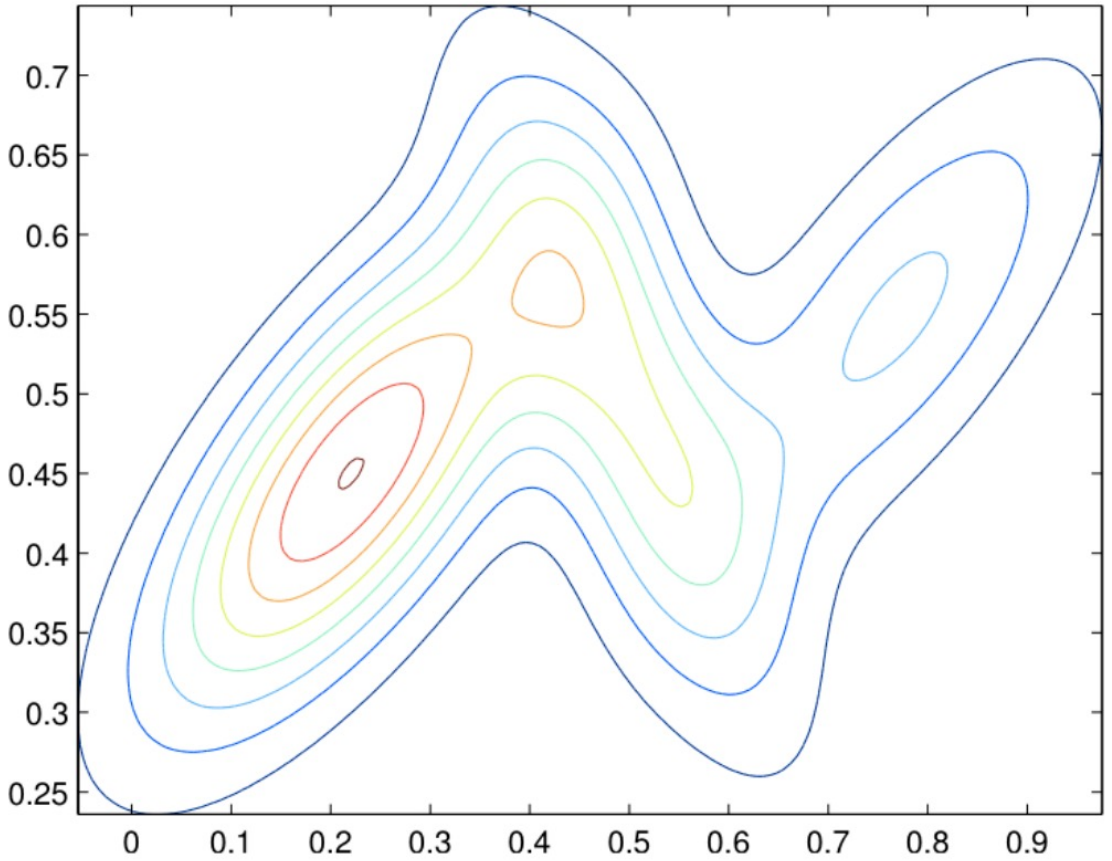
1. A GMM is typically considered as a density estimator. We will view it as a generative model.

2. GMMs are **universal approximators of densities** (if you have enough Gaussians). Even diagonal GMMs are universal approximators.

3. In general, mixture models are very powerful, but hard to optimize.

2D GMM Density Estimation

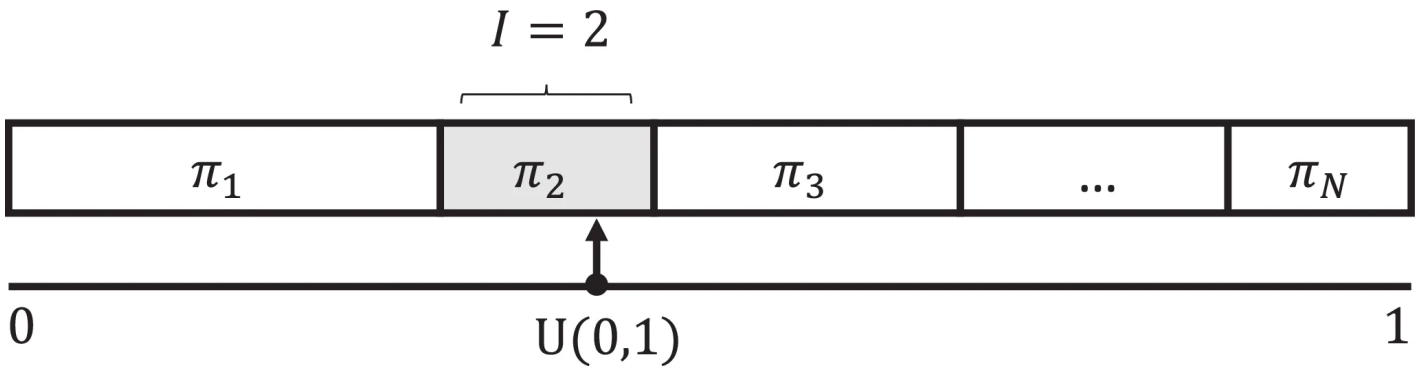




GMM Ancestral Sampling

1. Sample from categorical distribution (via inverse sampling theorem):

$$u \sim \mathcal{U}[0,1], \text{ choose } k \in \{1,...,K\} \text{ such that: } \frac{1}{k-1}\sum_{k'=0}^{k-1}\pi_k < u \leq \frac{1}{k}\sum_{k'=0}^{k}\pi_k.$$



2. Draw a Gaussian sample: $x \sim \mathcal{N}(\mu_k, \Sigma_k)$.

Fitting GMMs: Maximum Likelihood

- Dataset: $\mathcal{D} = \{x_1, \ldots, x_N\}$ model family: GMM.

- Likelihood function:

$$L(\theta; \mathcal{D}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}\left(x_n; \mu_k, \Sigma_k\right) \right).$$

Fitting GMMs: Maximum Likelihood

- Problems:
  1. **Singularities**: Arbitrarily large likelihood when a Gaussian component explains a single point.
  2. **Identifiability**: Solution is invariant to permutations (but not an issue if used as a generative model).

- How would you optimize this?
- Can we have a closed form update?
- Don't forget to satisfy the constraints on π_k and Σ_k.

Mixture Models as Latent Variable Model Marginalization

- We model the joint distribution as:

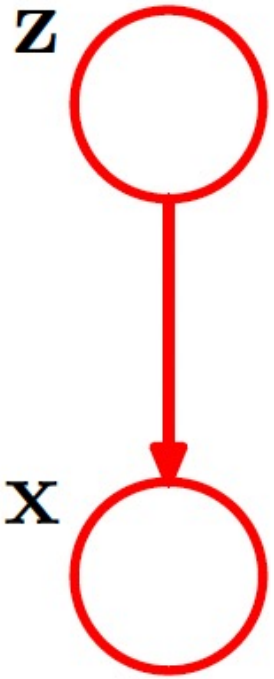
$$p(x, z) = p(x|z)p(z).$$

  $\hookrightarrow$ E.g., z indicates which component out of K is chosen.

- But we do not have the labels z. What can we do instead? Answer: Marginalize:

$$p(x) = \sum_z p(x, z) = \sum_z p(x|z)p(z).$$

  $\hookrightarrow$ This is a **mixture model**.



Latent Variable in GMMs

- In GMMs, a hidden (latent) variable z would represent which Gaussian generated our observation x, with some probability.

- Let $z \sim \text{Catergorical}(\pi)$, where $\pi_k \geq 0$, $\sum_{k=1}^{K} \pi_k = 1$. Then:

$$p(x) = \sum_{k=1}^{K} p(x, z = k) = \sum_{k=1}^{K} \underbrace{p(z = k)}_{\pi_k} \underbrace{p(x|z = k)}_{\mathcal{N}(x; \mu_k, \Sigma_k)}.$$

Latent Variable Models in General

- Some model variables may be unobserved, either at training or at test time, or both.
- If occasionally unobserved they are missing, e.g., undefined inputs, missing class labels, erroneous targets.
- Variables which are always unobserved are called **latent variables**, or sometimes **hidden variables**.

Back To GMMs

- A Gaussian mixture distribution: $p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}\left(x_n; \mu_k, \Sigma_k\right).$
- We had: $z \sim \text{Catergorical}(\pi)$, where $\pi_k \geq 0,\ \Sigma_k \pi_k = 1.$
- Join distribution: $p(x, z) = p(z)p(x|z).$
- Likelihood function:

$$L(\theta; \mathcal{D}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}\left(x_n; \mu_k, \Sigma_k\right) \right)$$

$$= \sum_{n=1}^{N} \log \left(\sum_{z_n=1}^{K} p\left(z_n; \pi\right) p\left(x_n | z_n; \mu, \Sigma\right) \right).$$

MLE for the Joint Distribution

- **If we knew** z_n for every x_n, the maximum likelihood problem is easy:

$$L(\pi, \mu, \Sigma; \mathcal{D}_{xz}) = \sum_{n=1}^{N} \log p\left(x_n, z_n; \pi, \mu, \Sigma\right)$$

$$= \sum_{n=1}^{N} \log p\left(x_n | z_n; \mu, \Sigma\right) + \log p\left(z_n; \pi\right).$$

Maximum Likelihood

- We would get for $k = 1, ..., K$:

$$\hat{\mu}_k = \frac{\sum_{n=1}^{N} 1_{[z_n=k]} x_n}{\sum_{n=1}^{N} 1_{[z_n=k]}},$$

$$\hat{\Sigma}_k = \frac{\sum_{n=1}^{N} 1_{[z_n=k]} \left(x_n - \hat{\mu}_k\right) \left(x_n - \hat{\mu}_k\right)^T}{\sum_{n=1}^{N} 1_{[z_n=k]}},$$

$$\hat{\pi}_k = \frac{1}{N} \sum_{n=1}^{N} 1_{[z_n=k]}.$$

How can we fit a GMM?

- Optimization uses the **Expectation Maximization (EM) algorithm**, which alternates between two steps:

  1. **E-step**: Compute the posterior probability over z given our current model, i.e., how much do we think each Gaussian generates each datapoint.

  2. **M-step**: Assuming that the data really was generated this way, change the parameters of each Gaussian to maximize the probability that it would generate the data it is currently responsible for.

     $\hookrightarrow$ We can derive closed form updates for all parameters.

Where does EM come from?

- Optimizing the likelihood is hard because of the sum inside of the log. Recalling $\theta := \{\pi_k, \mu_k, \Sigma_k, k = 1, ..., K\}$:

$$L(\theta; \mathcal{D}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k \mathcal{N}\left(x_n; \mu_k, \Sigma_k\right) \right) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} p\left(x_n, z_n = k; \theta\right) \right).$$

- Let's use a common trick in machine learning and introduce a new distribution, $q = [q_1, ..., q_K]^T$:

$$L(\theta; \mathcal{D}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} q_k \frac{p\left(x_n, z_n = k; \theta\right)}{q_k} \right).$$

Where does EM come from?

- Now we can swap them via *Jensen's inequality*! For the **concave** function, log, it holds:

$$g\left(\mathbb{E}[x]\right) = g\left(\sum_k p_k x_k\right) \geq \sum_k p_k g(x_k) = \mathbb{E}\left[g(x)\right].$$

- Applying Jensen's inequality:

$$L(\theta; \mathcal{D}) = \sum_{n=1}^{N} \log\left(\sum_{k=1}^{K} q_k \frac{p\left(x_n, z_n = k; \theta\right)}{q_k}\right) \geq \sum_{n=1}^{N}\sum_{k=1}^{K} q_k \log\left(\frac{p\left(x_n, z_n = k; \theta\right)}{q_k}\right).$$

- Maximizing this lower bound will force our likelihood to increase.
- But how do we pick a q_k that gives a good bound?

EM Derivation

- We got the sum outside, but we end up with an inequality:

$$L(\theta; \mathcal{D}) \geq \sum_{n=1}^{N} \sum_{k=1}^{K} q_k \log \left(\frac{p\left(x_n, z_n = k; \theta\right)}{q_k} \right).$$

- Let's fix the current parameters to θ^{old} and try to find a good q_k.

- What happens if we pick $q_{nk} = p\left(z_n = k | x_n; \theta^{\text{old}}\right)$? a posteriori probability of the latent variable given the sample

$\hookrightarrow \frac{p\left(x_n, z_n = k; \theta^{\text{old}}\right)}{p\left(z_n = k | x_n; \theta^{\text{old}}\right)} = p\left(x_n; \theta^{\text{old}}\right)$, and the inequality becomes an equality!

EM Derivation

- We can now define and optimize:

$$Q\left(\theta, \theta^{\text{old}}\right) := \sum_{n=1}^{N} \sum_{k=1}^{K} p\left(z_n = k | x_n; \theta^{\text{old}}\right) \log p\left(x_n, z_n = k; \theta\right)$$

$$= \sum_{n=1}^{N} \mathbb{E}_{p(z_n | x_n; \theta^{\text{old}})} \left[\log p\left(x_n, z_n; \theta\right)\right].$$

$\hookrightarrow$ We ignored the part that doesn't depend on θ.

EM Derivation

- So, what just happened?

- *Conceptually:* We don't know z_n so we average them given the current model.

- *Practically:* We define the function:

$$Q\left(\theta, \theta^{\text{old}}\right) := \sum_{n=1}^{N} \mathbb{E}_{p(z_n|x_n;\theta^{\text{old}})} \left[\log p\left(x_n, z_n; \theta\right)\right].$$

that lower bounds the desired function and is equal at our current guess.

EM Derivation

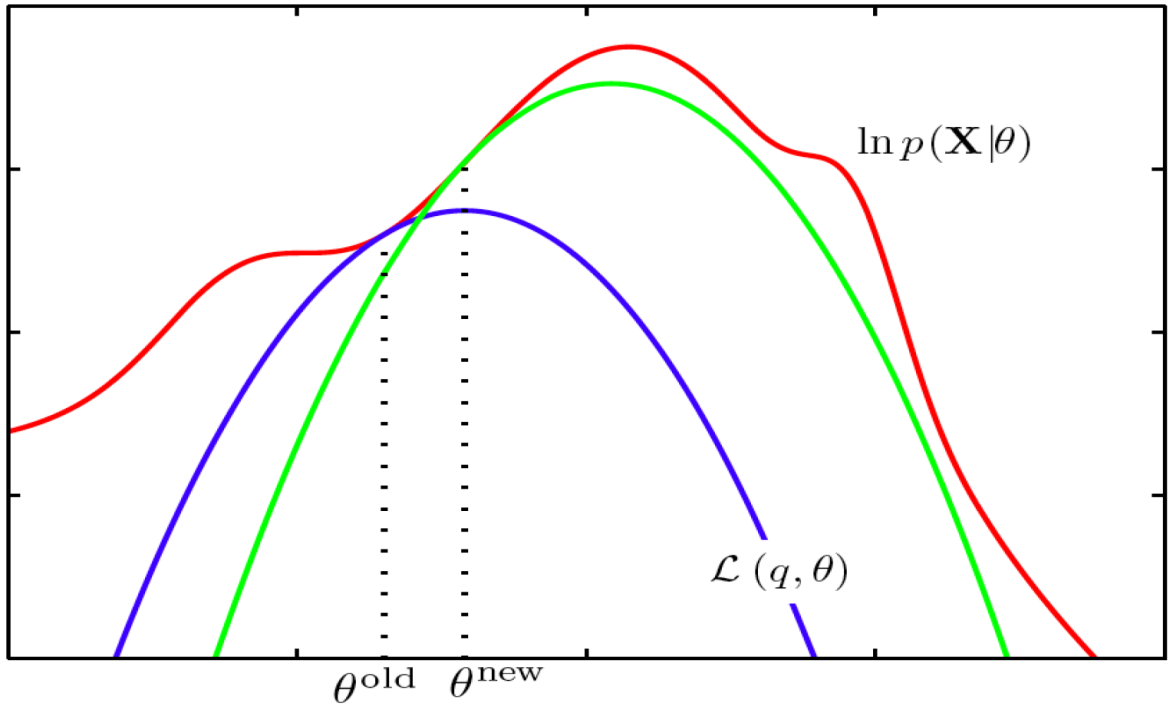
- Why it works?

- If we now optimize θ we will get a better lower bound!

$$L(\theta^{\text{old}}; \mathcal{D}) = Q\left(\theta^{\text{old}}, \theta^{\text{old}}\right) \leq Q\left(\theta^{\text{new}}, \theta^{\text{old}}\right) \leq L(\theta^{\text{new}}; \mathcal{D}).$$

- We can iterate between **expectation** step and **maximization** step and the lower bound will always improve (or we are done).

Visualization of the EM Algorithm



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

GMM E-Step: Responsibilities

- Using Bayes rule, the conditional a posteriori probability of z given x is written as:

$$\gamma_k = p\left(z = k|x\right) = \frac{p(z = k)p(x|z = k)}{p(x)}$$

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

- γ_k can be viewed as the responsibility of component k towards x.

GMM E-Step

- Once we computed $\gamma_{nk} = p\left(z_n = k | x_n; \theta^{\text{old}}\right)$ we can compute the expected log-likelihood:

$$\sum_{n=1}^{N} \mathbb{E}_{p(z_n|x_n;\theta^{\text{old}})} \left[\log\left(p\left(x_n, z_n; \theta\right)\right)\right]$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_{nk} \left[\log\left(p\left(z_n = k; \theta\right)\right) + \log\left(p\left(x_n | z_n = k; \theta\right)\right)\right]$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_{nk} \log\left(\pi_k\right) + \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_{nk} \log\left(\mathcal{N}\left(x_n; \mu_k, \Sigma_k\right)\right)$$

- To fit k Gaussians, we just need to weight each sample by γ_{nk}.

GMM M-Step

- Solving for μ_k and Σ_k is like fitting k separate Gaussians but with weights γ_{nk}. The solution is similar to what we have already seen:

$$\hat{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} x_n,$$

$$\hat{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} \left(x_n - \hat{\mu}_k\right) \left(x_n - \hat{\mu}_k\right)^T,$$

$$\hat{\pi}_k = \frac{N_k}{N}, \text{ with } N_k = \sum_{n=1}^{N} \gamma_{nk}.$$

effective sample size of cluster k

EM Algorithm for GMM

1. **Initialize** the means $\hat{\mu}_k^{(0)}$, covariances $\hat{\Sigma}_k^{(0)}$ and mixing coefficients $\hat{\pi}_k^{(0)}$.
2. Iterate $t = 0, 1, \ldots$ until convergence:
   2.1. **E-step**: Evaluate the responsibilities given the current parameters:

$$\gamma_{nk}^{(t)} = p\left(z_n = k | x_n\right) = \frac{\hat{\pi}_k^{(t)} \mathcal{N}\left(x_n; \hat{\mu}_k^{(t)}, \hat{\Sigma}_k^{(t)}\right)}{\sum_{j=1}^{K} \pi_j^{(t)} \mathcal{N}\left(x_n; \hat{\mu}_j^{(t)}, \hat{\Sigma}_j^{(t)}\right)}.$$

EM Algorithm for GMM

1. **Initialize** the means $\hat{\mu}_k^{(0)}$, covariances $\hat{\Sigma}_k^{(0)}$ and mixing coefficients $\hat{\pi}_k^{(0)}$.
2. Iterate $t = 0, 1, \ldots$ until convergence:

   2.2. **M-step**: Re-estimate the parameters given current responsibilities:

$$\hat{\mu}_k^{(t+1)} = \frac{1}{N_k^{(t)}} \sum_{n=1}^{N} \gamma_{nk}^{(t)} x_n, \qquad \hat{\pi}_k^{(t+1)} = \frac{N_k^{(t)}}{N}, \text{ with } N_k^{(t)} = \sum_{n=1}^{N} \gamma_{nk}^{(t)}.$$

$$\hat{\Sigma}_k^{(t+1)} = \frac{1}{N_k^{(t)}} \sum_{n=1}^{N} \gamma_{nk}^{(t)} \left(x_n - \hat{\mu}_k^{(t+1)}\right) \left(x_n - \hat{\mu}_k^{(t+1)}\right)^T.$$

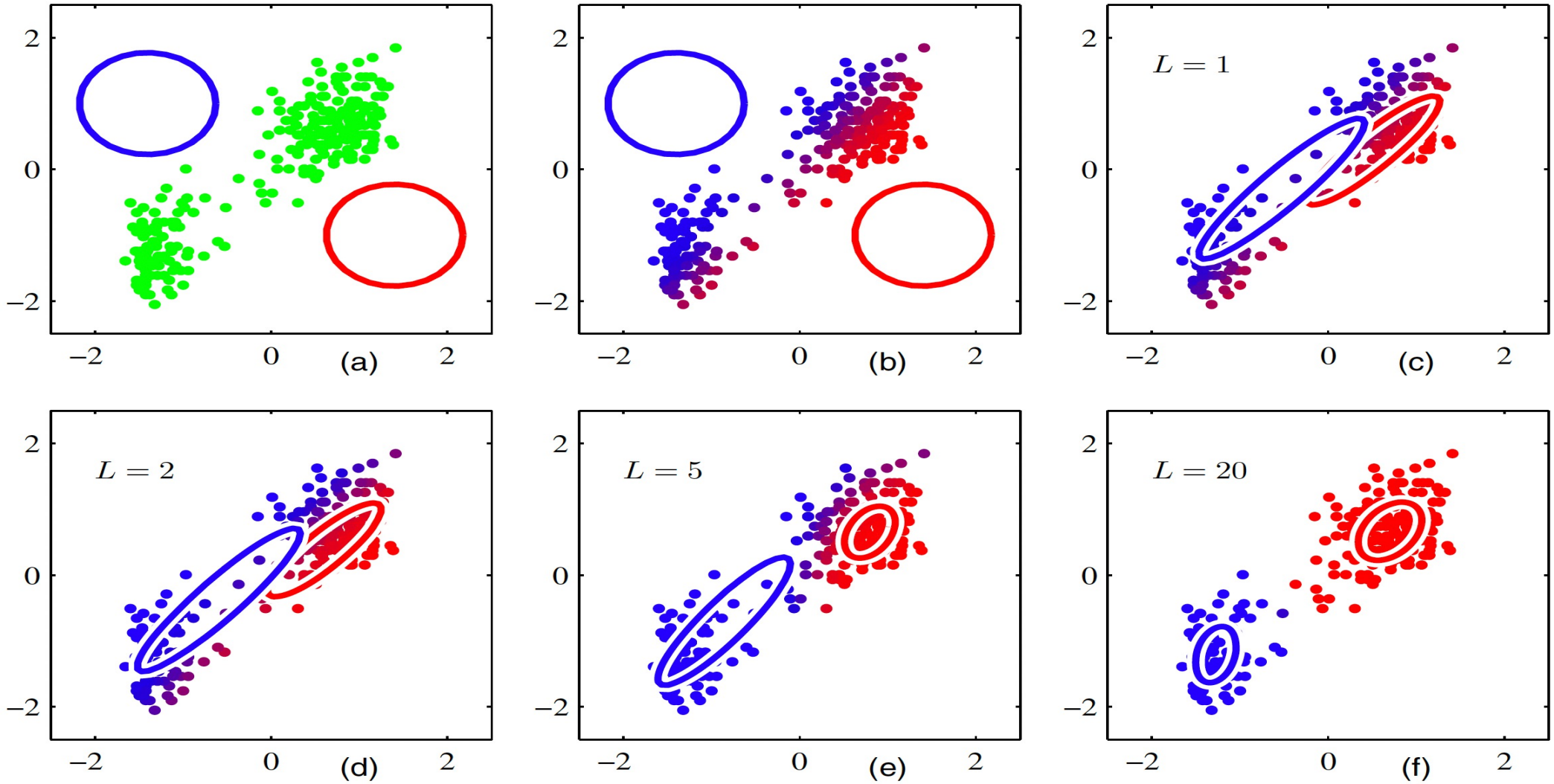
EM Algorithm for GMM

1. **Initialize** the means $\hat{\mu}_k^{(0)}$, covariances $\hat{\Sigma}_k^{(0)}$ and mixing coefficients $\hat{\pi}_k^{(0)}$.
2. Iterate $t = 0, 1, \ldots$ until convergence:

   2.3. Evaluate the *log-likelihood function* and check for convergence:

$$L(\hat{\theta}^{(t+1)}; \mathcal{D}) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} \hat{\pi}_k^{(t+1)} \mathcal{N}\left(x_n; \hat{\mu}_k^{(t+1)}, \hat{\Sigma}_k^{(t+1)}\right) \right).$$

Visualization of the EM Algorithm



General EM Algorithm

1. **Initialize** θ^{old}.

2. **E-step**: Evaluate $p\left(z|x;\theta^{\text{old}}\right)$ and compute:

$$Q\left(\theta,\theta^{\text{old}}\right)=\sum_{x}\sum_{z}p\left(z|x;\theta^{\text{old}}\right)\log p\left(x,z;\theta\right).$$

3. **M-step**: Maximize:

$$\theta^{\text{new}}=\arg\max_{\theta}Q\left(\theta,\theta^{\text{old}}\right).$$

4. Evaluate log-likelihood and check for convergence (or the parameters). If not converged, set $\theta^{\text{old}}=\theta^{\text{new}}$ and go to step 2.

Take home messages on GMMs & EM

1. GMM is a model that uses **latent variables** to represent its components.
2. Mixture models are very powerful models, **universal approximator**.
3. Optimization is performed using the **EM algorithm**.
4. EM is a general algorithm for optimizing many latent variable models.
5. EM iteratively computes a lower bound then optimizes it.
6. Converges but maybe to a local minima. So, we can use multiple restarts.
7. Limitation: need to be able to compute the posterior $p(z|x;\theta)$ which might not be possible for more complicated models.

$\hookrightarrow$ Solution: **Variational inference** (will be presented in VAEs).

References

1. Pattern Recognition and Machine Learning (*Chapter 9*)
   Christopher M. Bishop, Springer (2006)

2. "Maximum Likelihood from Incomplete Data via the EM Algorithm"
   Dempster, Laird and Rubin. Journal of the Royal Statistical Society,
   Series B. (1977)

3. `https://f.hubspotusercontent40.net/hubfs/8111846/Unicon_October2020/pdf/bilmes-em-algorithm.pdf`

4. `https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm`

Introduction to Deep Generative Modeling

Lecture #4

HY-673 – Computer Science Dep., University of Crete

Professors: Yannis Pantazis & Yannis Stylianou

TAs: Michail Raptakis & Michail Spanakis