# Bayesian Models in Machine Learning

GMM, EM algorithm

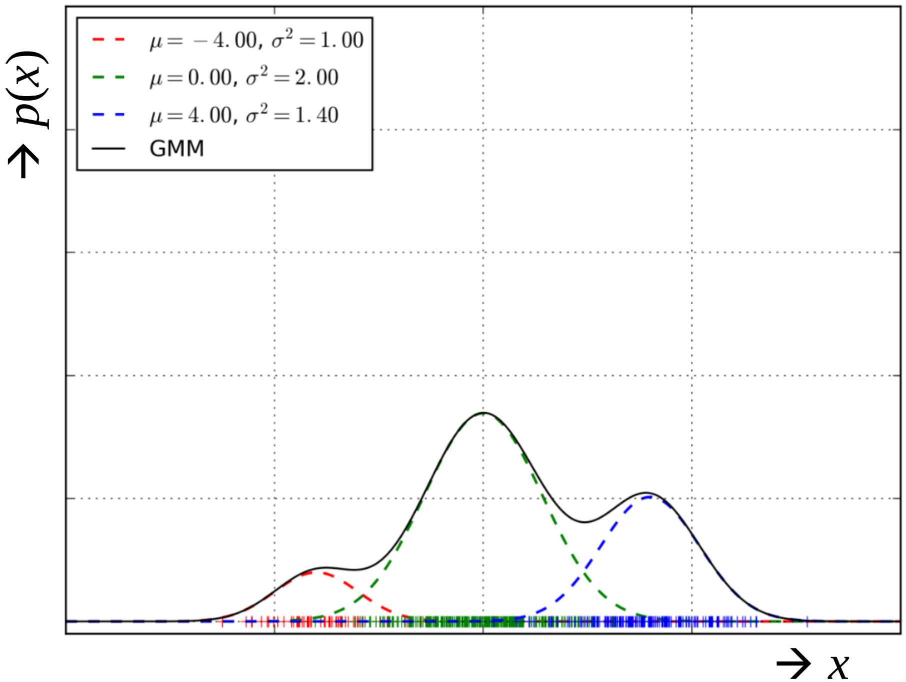

Lukáš Burget

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY


BAYa lectures, October 2023

# GMM - recapitulation

$$p(x|\boldsymbol{\eta}) = \sum_c \mathcal{N}(x; \mu_c, \sigma_c^2)\pi_c$$




where

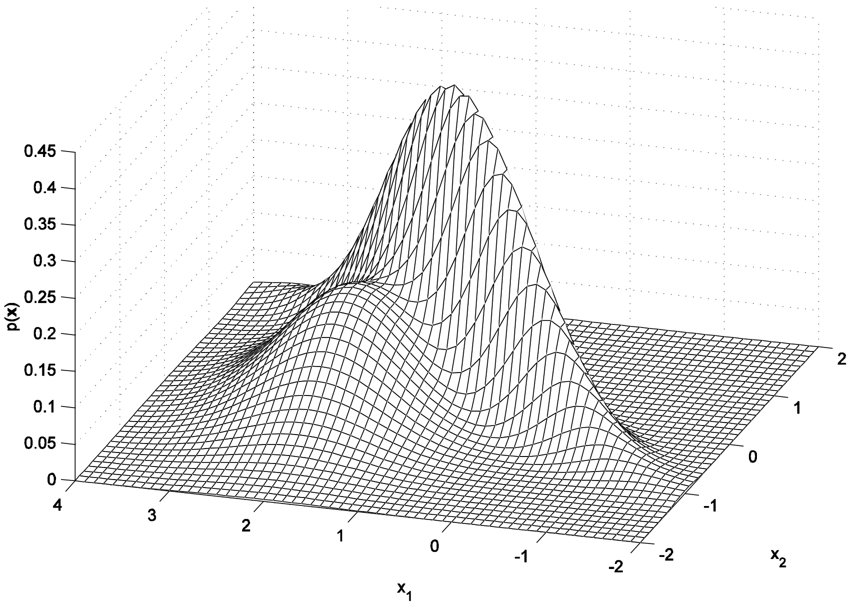
$$\boldsymbol{\eta} = \{\pi_c, \mu_c, \sigma_c^2\}$$

$$\sum_c \pi_c = 1$$

- We can see the sum above just as a function defining the shape of the probability density function
- or …

# Multivariate GMM - recapitulation

$$p(\mathbf{x}|\boldsymbol{\eta}) = \sum_c \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)\pi_c$$




where

$$\boldsymbol{\eta} = \{\pi_c, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c\}$$

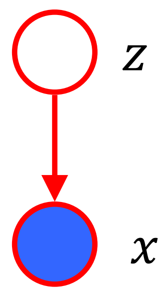
$$\sum_c \pi_c = 1$$

- We can see the sum above just as a function defining the shape of the probability density function
- or …

# BN for GMM – recapitulation

$$p(x) = \sum_z p(x|z)P(z) = \sum_c \mathcal{N}\left(x; \mu_c, \sigma_c^2\right)\mathrm{Cat}(z = c|\boldsymbol{\pi})$$

- or we can see it as a generative probabilistic model described by Bayesian network with **Categorical** latent random variable $z$ identifying **Gaussian** distribution generating the observation $x$

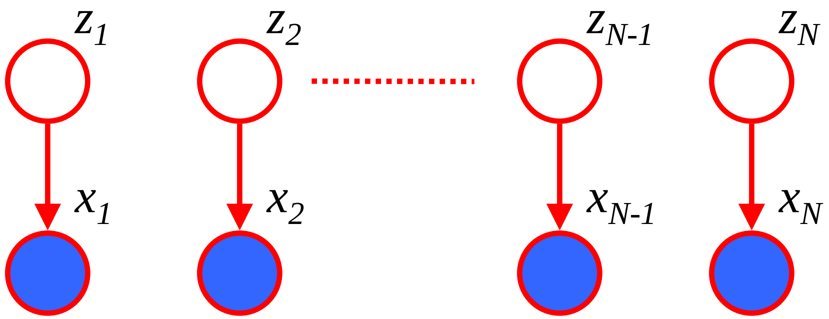



$$p(x, z) = p(x|z)P(z)$$

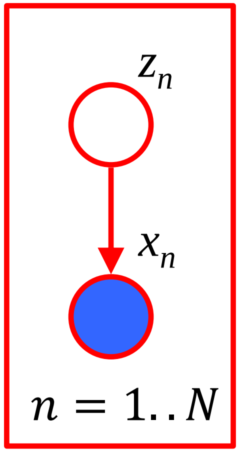
- Observations are assumed to be generated as follows:
  - randomly select Gaussian component according probabilities $P(z)$
  - generate observation $x$ form the selected Gaussian distribution
- To evaluate $p(x)$, we have to marginalize out $z$
- No close form solution for training

# BN for GMM – recapitulation II

- Multiple observations:



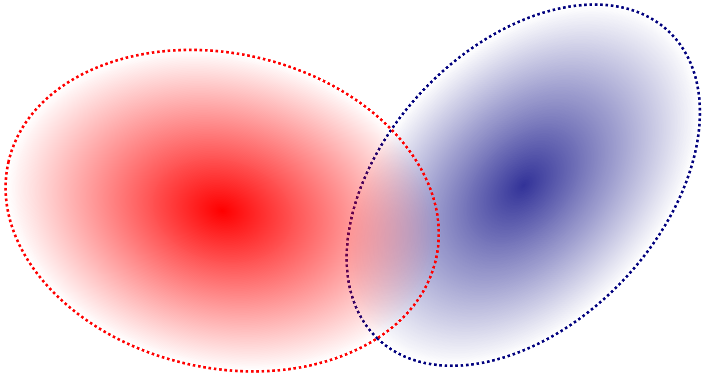

or




$$p(x_1, x_2, \dots, x_\mathrm{N}, z_1, z_2, \dots z_\mathrm{N}) = \prod_{n=1}^{N} p(x_\mathrm{n}|z_\mathrm{n}) P(z_n)$$

# Training GMM –Viterbi training

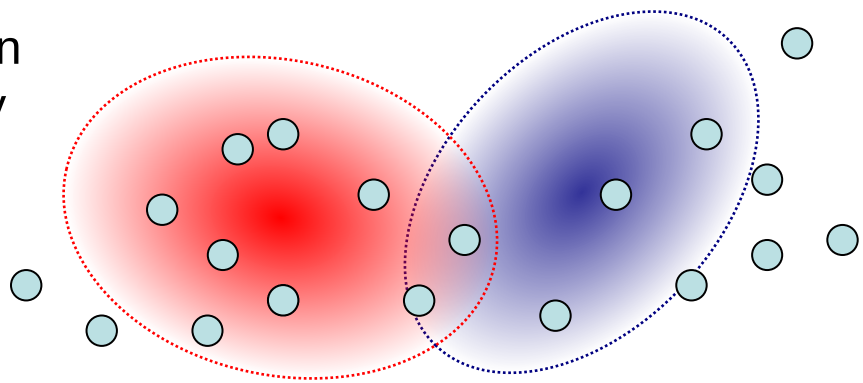
- Intuitive and Approximate iterative algorithm for training GMM parameters.



# Training GMM –Viterbi training

- Intuitive and Approximate iterative algorithm for training GMM parameters.

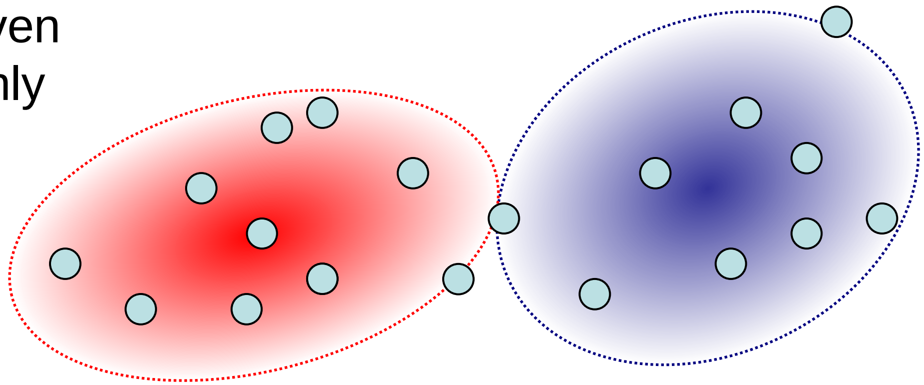
- Using current model parameters, let Gaussians classify data as if the Gaussians were different classes (Even though all the data corresponds to only one class modeled by the GMM)



# Training GMM –Viterbi training

- Intuitive and Approximate iterative algorithm for training GMM parameters.

- Using current model parameters, let Gaussians classify data as if the Gaussians were different classes (Even though all the data corresponds to only one class modeled by the GMM)



- Re-estimate parameters of Gaussians using the data assigned to them in the previous step. New weights will be proportional to the number of data points assigned to the Gaussians.

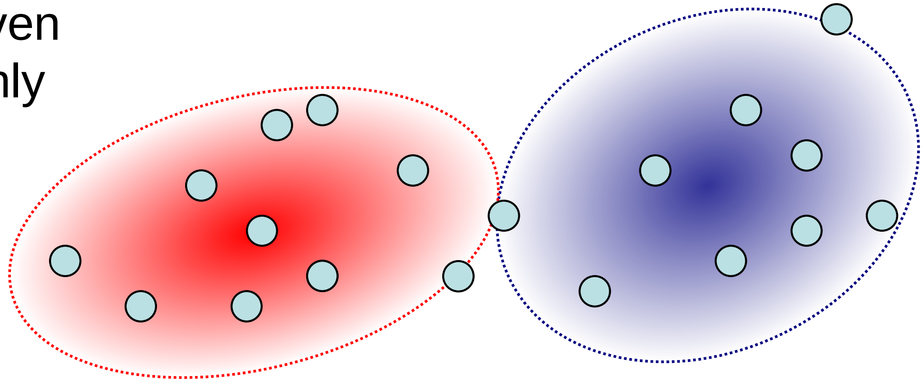
# Training GMM –Viterbi training

- Intuitive and Approximate iterative algorithm for training GMM parameters.

- Using current model parameters, let Gaussians classify data as if the Gaussians were different classes (Even though all the data corresponds to only one class modeled by the GMM)

- Re-estimate parameters of Gaussians using the data assigned to them in the previous step.
  New weights will be proportional to the number of  data points assigned to the Gaussians.

- Repeat the previous two steps until the algorithm converges.



# Training GMM – EM algorithm

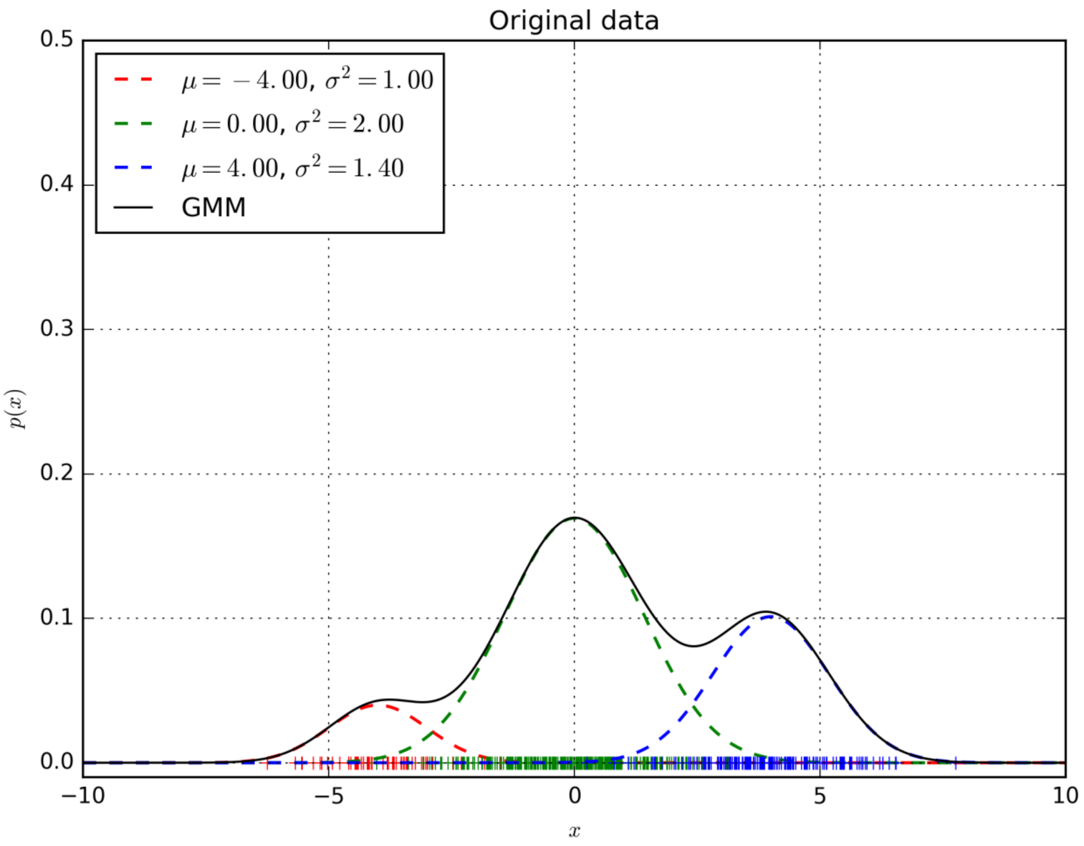
- **Expectation Maximization** is a general tool applicable to different generative models with latent (hidden) variables.
- Here, we only see the result of its application to the problem of re-estimating GMM parameters.
- It guarantees to increase the likelihood of training data in every iteration. However, it does not guarantee to find the global optimum.
- The algorithm is very similar to the Viterbi training presented above. However, instead of hard alignments of observations to Gaussian components, the posterior probabilities $P(c|x_i)$ (calculated given the old model) are used as soft weights. Parameters $\mu_c$,$\sigma_c^2$ are then calculated using a weighted average.

$$\gamma_{nc} = \frac{\mathcal{N}\left(x_n|\mu_c^{(old)}, {\sigma^2}_c^{(old)}\right)\pi_c^{(old)}}{\sum_k \mathcal{N}\left(\mathrm{x}_n|\mu_k^{(old)}, {\sigma^2}_k^{(old)}\right)\pi_k^{(old)}} = \frac{p(\mathrm{x}_n|z_n = c)P(z_n = c)}{\sum_k p(x_n|z_n = k)P(z_n = k)} = P(z_n = c|\mathrm{x}_n)$$

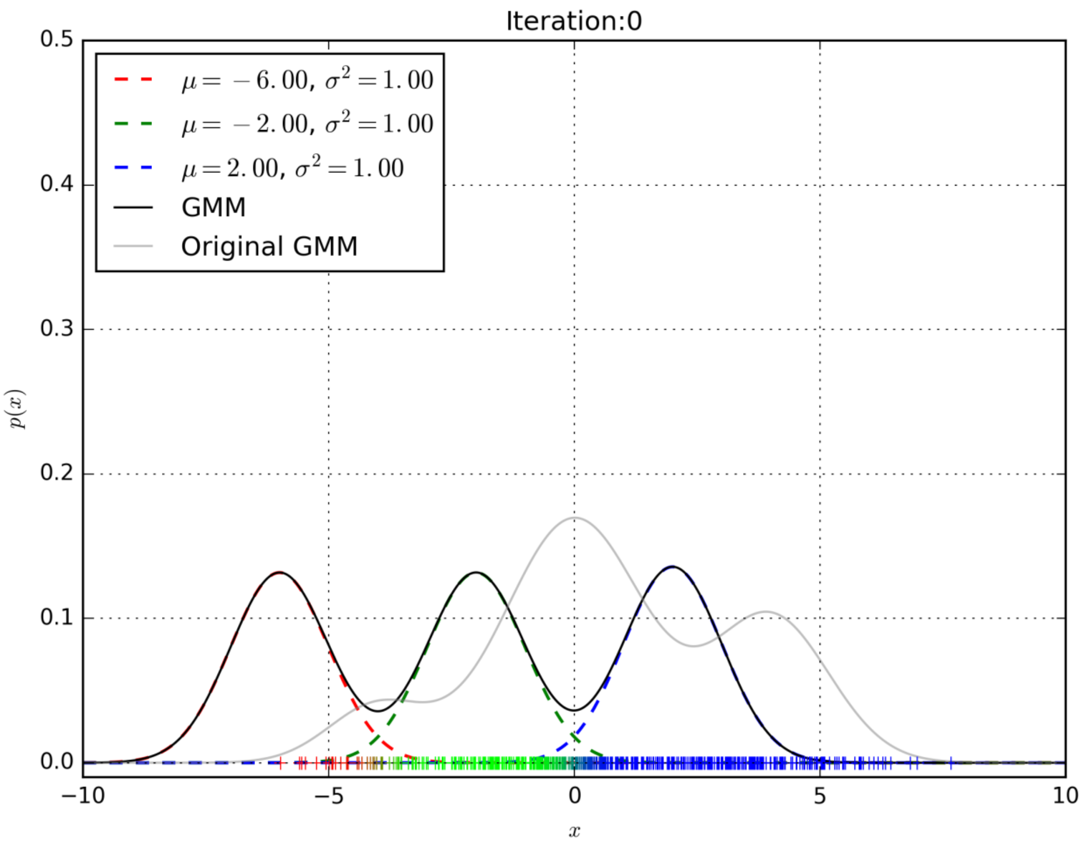
$$\mu_c^{(new)} = \frac{1}{\sum_n \gamma_{nc}} \sum_n \gamma_{nc}\mathrm{x}_n \qquad \pi_c^{(new)} = \frac{\sum_n \gamma_{nc}}{\sum_k \sum_n \gamma_{nc}} = \frac{\sum_n \gamma_{nc}}{N}$$

$${\sigma^2}_c^{(new)} = \frac{1}{\sum_n \gamma_{nc}} \sum_n \gamma_{nc}\left(\mathrm{x}_\mathrm{n} - \mu_c^{(new)}\right)^2$$

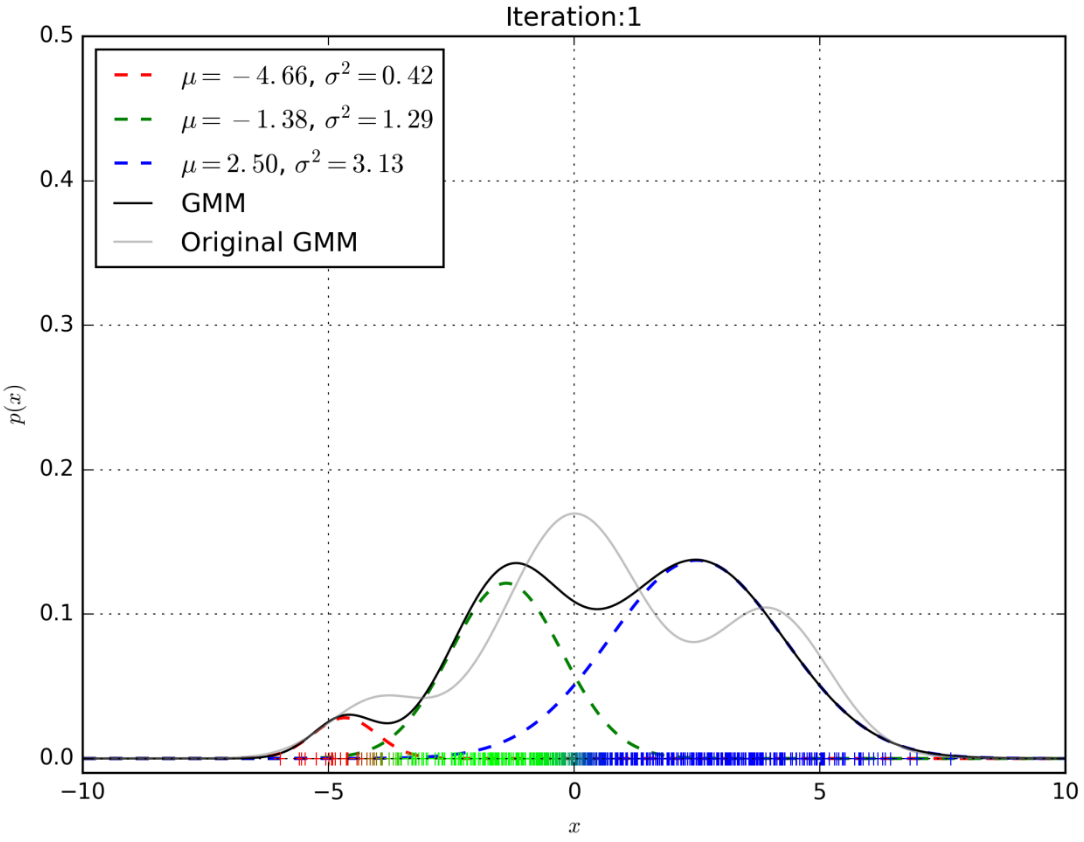
# GMM to be learned



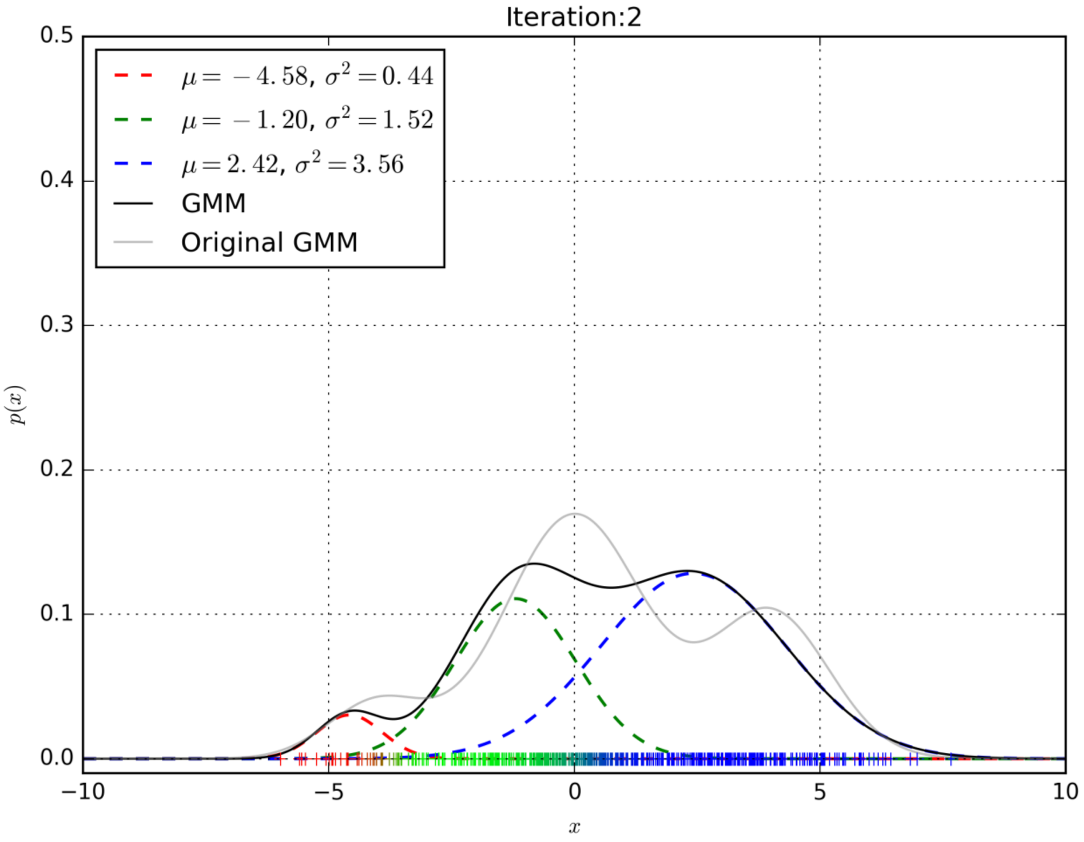
# EM algorithm



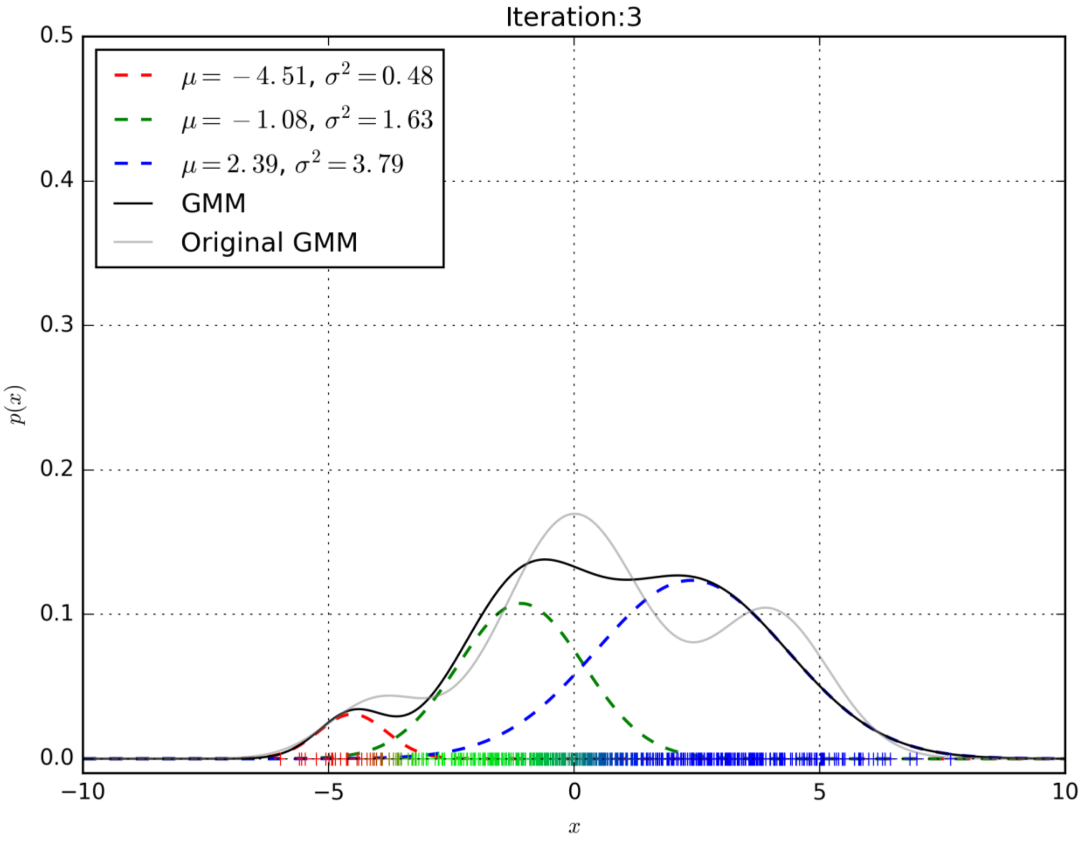
# EM algorithm



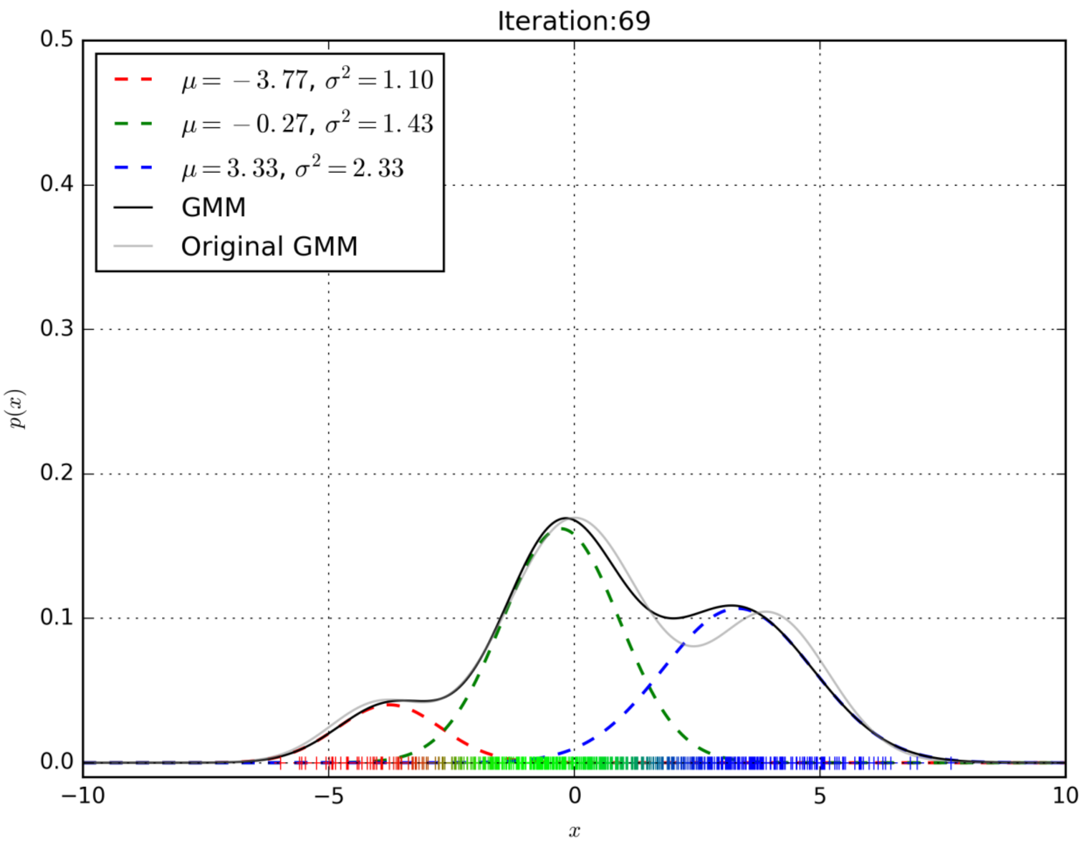
# EM algorithm



# EM algorithm



# EM algorithm



# Expectation maximization algorithm

$$\ln p(\mathbf{X}|\boldsymbol{\eta}) = \underbrace{\left(\sum_{\mathbf{Z}} q(\mathbf{Z})\right)}_{=1} \ln \underbrace{\frac{p(\mathbf{X},\mathbf{Z}|\boldsymbol{\eta})}{p(\mathbf{Z}|\mathbf{X},\boldsymbol{\eta})}}_{=p(\mathbf{X}),\forall\mathbf{Z}} \underbrace{\frac{q(\mathbf{Z})}{q(\mathbf{Z})}}_{=1} = \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{X},\mathbf{Z}|\boldsymbol{\eta})q(\mathbf{Z})}{p(\mathbf{Z}|\mathbf{X},\boldsymbol{\eta})q(\mathbf{Z})}$$

$$= \underbrace{\underbrace{\sum_{\mathbf{Z}} q(\mathbf{Z}) \ln p(\mathbf{X},\mathbf{Z}|\boldsymbol{\eta})}_{\mathcal{Q}(q(\mathbf{Z}),\boldsymbol{\eta})} \underbrace{- \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln q(\mathbf{Z})}_{H(q(\mathbf{Z}))}}_{\mathcal{L}(q(\mathbf{Z}),\boldsymbol{\eta})} \underbrace{- \sum_{\mathbf{Z}} q(\mathbf{Z}) \ln \frac{p(\mathbf{Z}|\mathbf{X},\boldsymbol{\eta})}{q(\mathbf{Z})}}_{D_{KL}(q(\mathbf{Z})||p(\mathbf{Z}|\mathbf{X},\boldsymbol{\eta}))}$$

- where $q(\mathbf{Z})$ is any distribution over the latent variable
- Kullback-Leibler divergence $\mathrm{D}_{\mathrm{KL}}(q||p)$ measures “unsimilarity” between two distributions $q, p$
- $D_{KL}(q||p) \geq 0$ and $D_{\mathrm{KL}}(q||p) = 0 \Leftrightarrow q = p$
- $\Rightarrow$ Evidence lower bound (**ELBO**) $\mathcal{L}(q(\mathbf{Z}), \boldsymbol{\eta}) \leq p(\mathbf{X}|\boldsymbol{\eta})$
- $H(q(\mathbf{Z}))$ is (non-negative) Entropy of distribution $q(\mathbf{Z})$
- $\mathcal{Q}(q(\mathbf{Z}), \boldsymbol{\eta})$ is called auxiliary function.

# Expectation maximization algorithm

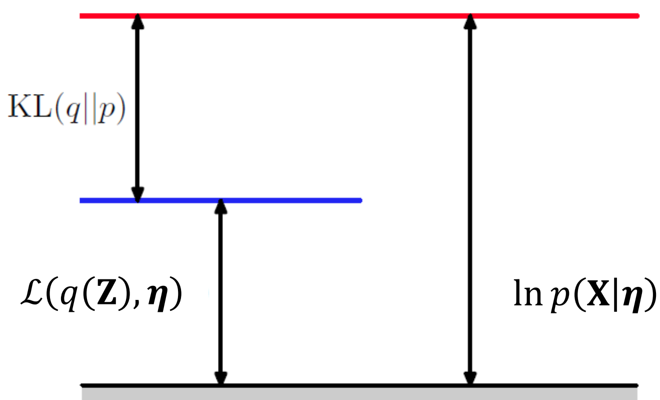
$$\ln p(\mathbf{X}|\boldsymbol{\eta}) = \underbrace{\mathcal{Q}(q(\mathbf{Z}), \boldsymbol{\eta}) + H(q(\mathbf{Z}))}_{\mathcal{L}(q(\mathbf{Z}),\boldsymbol{\eta})} + D_{KL}\left(q(\mathbf{Z})||p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\eta})\right)$$

- We aim to find parameters $\boldsymbol{\eta}$ that maximize $\ln p(\mathbf{X}|\boldsymbol{\eta})$
- E-step: $q(\mathbf{Z}) \coloneqq P(\mathbf{Z}|\mathbf{X}, \boldsymbol{\eta}^{old})$
  - makes the $D_{\mathrm{KL}}(q||p)$ term 0
  - makes $\mathcal{L}(q(\mathbf{Z}), \boldsymbol{\eta}) = \ln p(\mathbf{X}|\boldsymbol{\eta})$

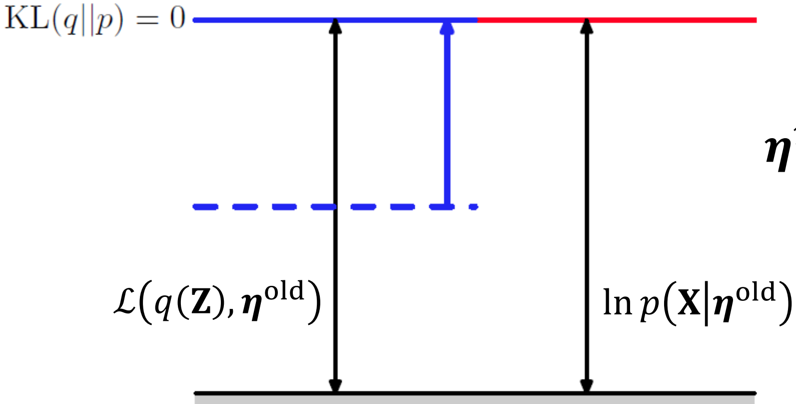
- M-step: $\boldsymbol{\eta}^{new} = \arg\max_{\boldsymbol{\eta}} \mathcal{Q}(q(\mathbf{Z}), \boldsymbol{\eta})$
  - $D_{KL}(q||p)$ increases as $P(\mathbf{X}|\mathbf{Z}, \boldsymbol{\eta})$deviates from $q(\mathbf{Z})$
  - $H\big(q(\mathbf{Z})\big)$ does not change for fixed $q(\mathbf{Z})$
  - $\mathcal{L}(q(\mathbf{Z}), \boldsymbol{\eta})$ increases like $\mathcal{Q}(q(\mathbf{Z}), \boldsymbol{\eta})$
  - $\ln p(\mathbf{X}|\boldsymbol{\eta})$ increases more than $\mathcal{Q}(q(\mathbf{Z}), \boldsymbol{\eta})$

# Expectation maximization algorithm

$$\ln p(\mathbf{X}|\boldsymbol{\eta}) = \underbrace{\mathcal{Q}(q(\mathbf{Z}),\boldsymbol{\eta}) + H(q(\mathbf{Z}))}_{\mathcal{L}(q(\mathbf{Z}),\boldsymbol{\eta})} + D_{KL}\left(q(\mathbf{Z})||p(\mathbf{Z}|\mathbf{X},\boldsymbol{\eta})\right)$$

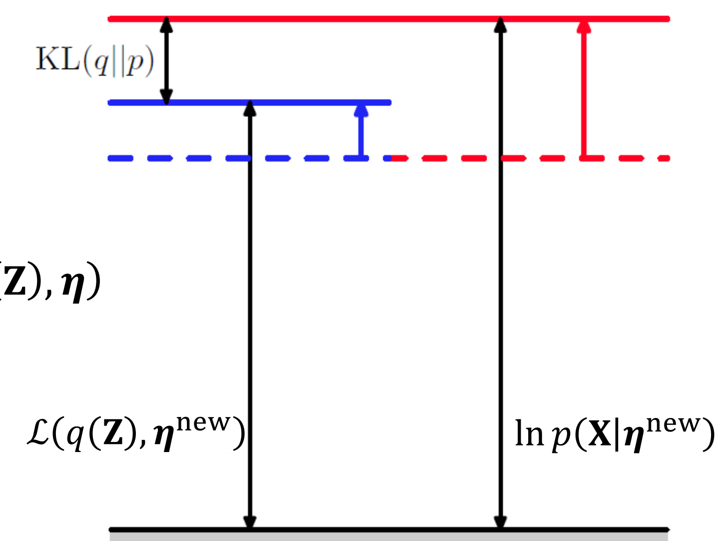



⇩ E-step: $q(\mathbf{Z}) \coloneqq P(\mathbf{Z}|\mathbf{X},\boldsymbol{\eta}^{old})$



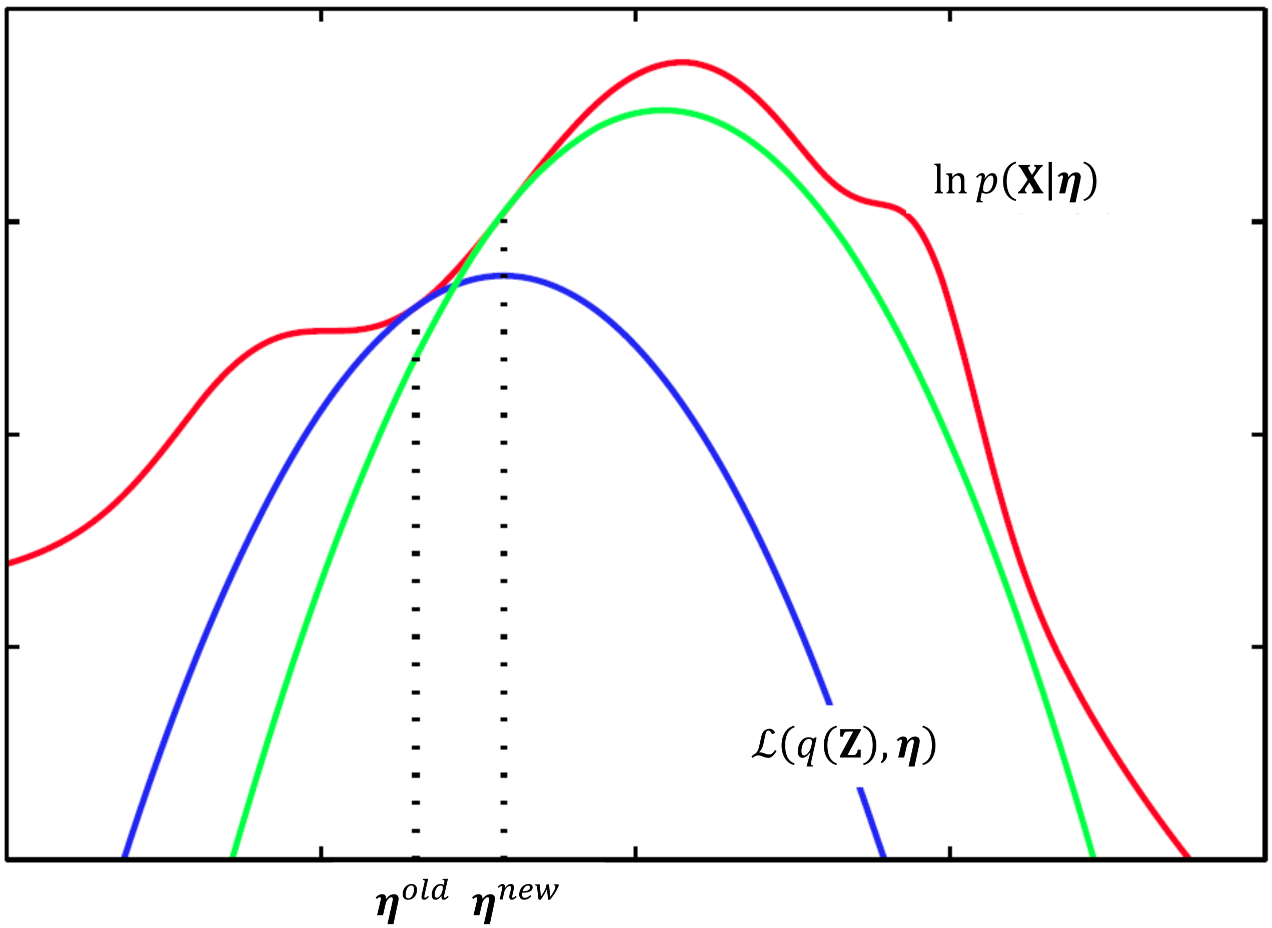

M-step: ⇨

$$\boldsymbol{\eta}^{new} = \arg\max_{\boldsymbol{\eta}} \mathcal{Q}(q(\mathbf{Z}),\boldsymbol{\eta})$$



# Expectation maximization algorithm

$\mathcal{Q}(q(\mathbf{Z}), \boldsymbol{\eta})$ and $\mathcal{L}(q(\mathbf{Z}), \boldsymbol{\eta})$ will be easy to optimize (e.g. quadratic function) compared to $\ln p(\mathbf{X}|\boldsymbol{\eta})$



# EM for GMM

- Now, we aim to train parameters $\boldsymbol{\eta} = \{\mu_z, \sigma_z^2, \pi_z\}$ of Gaussian Mixture model

$$p(x) = \sum_z p(x|z)P(z) = \sum_c \mathcal{N}(x; \mu_c, \sigma_c^2)\mathrm{Cat}(z = c|\boldsymbol{\pi})$$

- Given training observations $\mathbf{x} = [x_1, x_2, \dots, x_N]$ we search for ML estimate of $\boldsymbol{\eta}$ that maximizes log likelihood of the training data.

$$\ln p(\mathbf{x}) = \sum_n \ln p(x_n) = \sum_n \left[\ln \sum_c \mathcal{N}(x_n; \mu_c, \sigma_c^2) \qquad \pi_c\right]$$

- Direct maximization of this objective function w.r.t. $\boldsymbol{\eta}$ is intractable.
- We will use EM algorithm, where we maximize the auxiliary function which is (for simplicity) sum of per-observation auxiliary functions

$$Q(q(\mathbf{z}), \boldsymbol{\eta}) = \sum_n Q_n(q(z_n), \boldsymbol{\eta})$$

- Again, in M-step $\sum_n \ln p(x_n)$ has to increase more than $\sum_n Q_n(q(z_n), \boldsymbol{\eta})$

# EM for GMM – E-step

$$q(z_n) = P(z_n|x_n, \boldsymbol{\eta}^{old})$$
$$= \frac{p(x_n|z_n, \boldsymbol{\eta}^{old})P(z_n|\boldsymbol{\eta}^{old})}{p(x_n|\boldsymbol{\eta}^{old})}$$

$$q(z_n = c) = \frac{\mathcal{N}(x_n|\mu_c^{old}\sigma_c^{2^{old}})\pi_c^{old}}{\sum_k \mathcal{N}(x_n|\mu_k^{old}, \sigma_k^{2^{old}})\pi_k^{old}}$$
$$= \gamma_{nc}$$

- $\gamma_{nc}$ is the so called responsibility of Gaussian component $z$ for observation $n$.
- It is the probability for an observation $n$ being generated from component $c$

# EM for GMM – M-step

$$
\begin{aligned}
Q(q(\mathbf{z}), \boldsymbol{\eta}) &= \sum_n Q_n(q(z_n), \boldsymbol{\eta}) \\
&= \sum_n \sum_{z_n} q(z_n) \ln p(x_n, z_n | \boldsymbol{\eta}) \\
&= \sum_n \sum_c \gamma_{nc} \left[\ln \mathcal{N}(x_n; \mu_c, \sigma_c) + \ln \pi_c\right]
\end{aligned}
$$

- In M-step, the auxiliary function is maximized w.r.t. all GMM parameters

# EM for GMM –update of means

- Update for component mean means:

$$\frac{\partial}{\partial \mu_c} \sum_n Q_n(q(z_n), \boldsymbol{\eta}) = \frac{\partial}{\partial \mu_c} \sum_n \sum_k \gamma_{nk} \left[\ln \mathcal{N}(x_n; \mu_k, \sigma_k^2) + \ln \pi_k\right]$$

$$= \frac{\partial}{\partial \mu_c} \sum_n \gamma_{nc} \left[ - \frac{(x_n - \mu_c)^2}{2\sigma_c^2} + K \right]$$

$$= \frac{1}{\sigma_c^2} \sum_n \gamma_{nc} (\mu_c - x_n) = 0$$

$$\Longrightarrow \mu_c = \frac{\sum_n \gamma_{nc} x_n}{\sum_n \gamma_{nc}}$$

- Update for variances: $\sigma_c^2 = \frac{\sum_n \gamma_{nc} (\mathrm{x_n} - \mu_c)^2}{\sum_n \gamma_{nc}}$ can be derived similarly.

# Flashback: ML estimate for Gaussian

$$\arg\max_{\mu,\sigma^2} p(\mathbf{x}|\mu,\sigma^2) = \arg\max_{\mu,\sigma^2} \ln p(\mathbf{x}|\mu,\sigma^2) = \sum_i \ln \mathcal{N}(x_n;\mu,\sigma^2)$$

$$= -\frac{1}{2\sigma^2}\sum_n x_n^2 + \frac{\mu}{\sigma^2}\sum_n x_n - N\frac{\mu^2}{2\sigma^2} - \frac{\ln(2\pi)}{2}$$

$$\frac{\partial}{\partial\mu}\ln p(\mathbf{x}|\mu,\sigma^2) = \frac{\partial}{\partial\mu}\left(-\frac{1}{2\sigma^2}\sum_n x_n^2 + \frac{\mu}{\sigma^2}\sum_n x_n - N\frac{\mu^2}{2\sigma^2} - \frac{\ln(2\pi)}{2}\right)$$

$$= \frac{1}{\sigma^2}\left(\sum_n x_n - N\mu\right) = 0 \quad \Rightarrow \quad \hat{\mu}^{ML} = \frac{1}{N}\sum\nolimits_n x_n$$

and similarly: $\widehat{\sigma^2}^{ML} = \frac{1}{N}\sum\nolimits_n (x_n-\mu)^2$

# EM for GMM –update of weights

- Weights $\pi_c$ need to sum up to one. When updating weights, Lagrange multiplier $\lambda$ is used to enforce this constraint.

$$\frac{\partial}{\partial \pi_c} \left( \sum_n Q_n(q(z_n), \boldsymbol{\eta}) - \lambda \left( \sum_k \pi_k - 1 \right) \right) =$$

$$\frac{\partial}{\partial \pi_c} \left( \sum_n \sum_k \gamma_{nk} \ln \pi_k - \lambda \left( \sum_k \pi_k - 1 \right) \right) =$$

$$\sum_n \frac{\gamma_{nc}}{\pi_c} - \lambda = 0$$

$$\Longrightarrow \pi_c = \frac{\sum_n \gamma_{nc}}{\lambda} = \frac{\sum_n \gamma_{nc}}{\sum_k \sum_n \gamma_{nk}}$$

# Factorization of the auxiliary function more formally

- Before, we have introduced the per-observation auxiliary functions

$$Q(q(\mathbf{z}), \boldsymbol{\eta}) = \sum_n Q_n(q(z_n), \boldsymbol{\eta})$$
$$= \sum_n \sum_{z_n} q(z_n) \ln p(x_n, z_n | \boldsymbol{\eta})$$

- We can show that such factorization comes naturally even if we directly write the auxiliary function as defined for the EM algorithm:

$$\mathcal{Q}(q(\mathbf{z}), \boldsymbol{\eta}) = \sum_{\mathbf{z}} q(\mathbf{z}) \ln p(\mathbf{x}, \mathbf{z} | \boldsymbol{\eta}) = \sum_{\mathbf{z}} \prod_{n'} q(z_{n'}) \sum_n \ln p(x_n, z_n | \boldsymbol{\eta})$$
$$= \sum_c \sum_n q(z_n = c) \ln p(x_n, z_n = c | \boldsymbol{\eta})$$

- See the next slide for proof

# Factorization over components

Example with only 3 observations (i.e., $\mathbf{z} = [\mathrm{z}_1, \mathrm{z}_2, \mathrm{z}_3]$)

$$\sum_{\mathbf{z}} q(\mathbf{z}) \ln p(\mathbf{x}, \mathbf{z}|\boldsymbol{\eta}) = \sum_{\mathbf{z}} \prod_{n'} q(z_{n'}) \sum_{n} \log p(x_n, z_n|\boldsymbol{\eta}) = \sum_{\mathbf{z}} \prod_{n'} q(z_{n'}) \sum_{n} f(z_n) = \sum_{n} \sum_{\mathbf{z}} \prod_{n'} q(z_{n'}) f(z_n) =$$

$$\sum_{z_1} \sum_{z_2} \sum_{z_3} q(z_1) q(z_2) q(z_3) f(z_1) + \sum_{z_1} \sum_{z_2} \sum_{z_3} q(z_1) q(z_2) q(z_3) f(z_2) + \sum_{z_1} \sum_{z_2} \sum_{z_3} q(z_1) q(z_2) q(z_3) f(z_3) =$$

$$\sum_{z_1} q(z_1) f(z_1) \sum_{z_2} q(z_2) \sum_{z_3} q(z_3) + \sum_{z_1} q(z_1) \sum_{z_2} q(z_2) f(z_2) \sum_{z_3} q(z_3) + \sum_{z_1} q(z_1) \sum_{z_2} q(z_2) \sum_{z_3} q(z_3) f(z_3) =$$

$$\sum_{z_1} q(z_1) f(z_1) + \sum_{z_2} q(z_2) f(z_2) + \sum_{z_3} q(z_3) f(z_3) =$$

$$\sum_{c=1}^{C} q(z_1 = c) f(z_1 = c) + \sum_{c=1}^{C} q(z_2 = c) f(z_2 = c) + \sum_{c=1}^{C} q(z_3 = c) f(z_3 = c) =$$

$$\sum_{c=1}^{C} \sum_{n} q(z_n = c) f(z_n = c) = \sum_{c=1}^{C} \sum_{n} q(z_n = c) \log p(x_n, z_n = c|\eta)$$

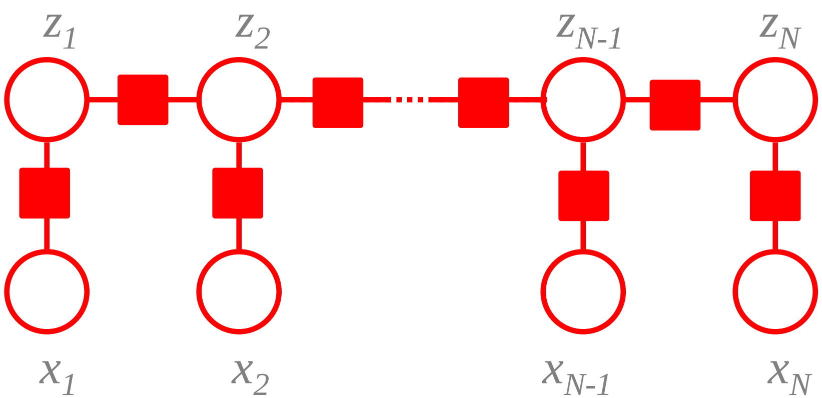
# Flashback: Example: BP for HMM

- To evaluation an HMM, given a sequence of observations $\boldsymbol{X} = [x_1, x_2 \dots, x_N]$, we need to infer

$$p(\boldsymbol{X}) = p(x_1, x_2 \dots, x_N) = \sum_{z_1}\sum_{z_2} \dots \sum_{z_N} p(x_1, x_2 \dots, x_N, z_1, z_2 \dots, z_N)$$

- To train an HMM using an EM algorithm (see next lesson), for every $t = 1..N$, we need to infer

$$p(z_t|\boldsymbol{X}) = \frac{p(z_t, \boldsymbol{X})}{p(\boldsymbol{X})} = \frac{\sum_{z_1}\sum_{z_2} \dots \sum_{z_{t-1}}\sum_{z_{t+1}} \dots \sum_{z_N} p(x_1, x_2 \dots, x_N, z_1, z_2 \dots, z_N)}{p(\boldsymbol{X})}$$




***Forward-backward algorithm***

*$s$ are state ids (i.e., possible values of $z_t$)*

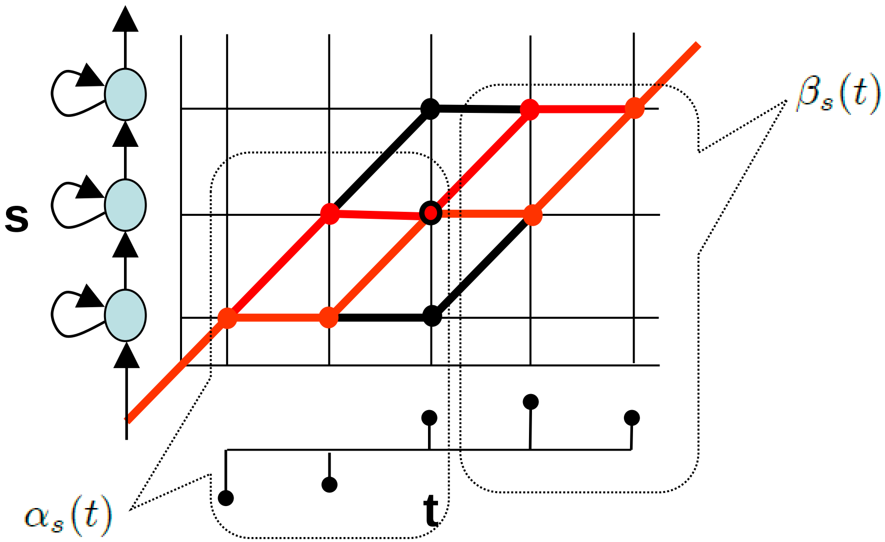
$$\alpha(t, s) = p(\mathbf{x}_t|s)\sum_{s'} \alpha(t-1, s')p(s|s')$$

$$\beta(t, s) = \sum_{s'} \beta(t+1, s')p(\mathbf{x}_{t+1}|s')p(s'|s)$$

$$p(\boldsymbol{X}) = \sum_{s' \in FinalStates} \alpha(N, s')$$

$$p(z_t = s|\boldsymbol{X}) = \frac{\alpha(t, s)\beta(t, s)}{P(\mathbf{X})}$$

# Examples: Training HMMs using EM




*E-step:*

$$\alpha(t,s) = p(\mathbf{x}_t|s)\sum_{s'}\alpha(t-1,s')p(s|s')$$

$$\beta(t,s) = \sum_{s'}\beta(t+1,s')p(\mathbf{x}_{t+1}|s')p(s'|s)$$

$$\gamma_s(t) = p(z_t = s|\boldsymbol{X}) = \frac{\alpha(t,s)\beta(t,s)}{\sum_{s'\in FinalStates}\alpha(N,s')}$$

*M-step:*

$$\hat{\mu}_s^{(new)} = \frac{\sum_{t=1}^{T}\gamma_s(t)x(t)}{\sum_{t=1}^{T}\gamma_s(t)}$$

$$\hat{\sigma}_s^{2(new)} = \frac{\sum_{t=1}^{T}\gamma_s(t)(x(t)-\hat{\mu}_s^{(new)})^2}{\sum_{t=1}^{T}\gamma_s(t)}$$

# EM for continuous latent variable

- Same equations, where sums over the latent variable $\mathbf{Z}$ are simply replaced by integrals

$$\ln p(\mathbf{X}|\boldsymbol{\eta}) = \underbrace{\int q(\mathbf{Z}) \ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\eta})\, \mathrm{d}\mathbf{Z}}_{\mathcal{Q}(q(\mathbf{Z}),\boldsymbol{\eta})} \underbrace{- \int q(\mathbf{Z}) \ln q(\mathbf{Z})\, \mathrm{d}\mathbf{Z}}_{H(q(\mathbf{Z}))} \underbrace{- \int q(\mathbf{Z}) \ln \frac{p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\eta})}{q(\mathbf{Z})}\, \mathrm{d}\mathbf{Z}}_{D_{KL}(q(\mathbf{Z})||p(\mathbf{Z}|\mathbf{X},\boldsymbol{\eta}))}$$

$$= \underbrace{\mathcal{Q}(q(\mathbf{Z}), \boldsymbol{\eta}) + H(q(\mathbf{Z}))}_{\mathcal{L}(q(\mathbf{Z}),\boldsymbol{\eta})} + D_{KL}\left(q(\mathbf{Z})||p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\eta})\right)$$

# Flashback: PLDA model for speaker verification

- Let each speech utterance be represented by *speaker embedding vector* $\mathbf{x}$
  - e.g. 512 dim. output of hidden layer of neural network trained for speaker classification
- We assume, that the distribution of the embeddings can be modeled as follows:
- We assume the same factorization as for GMM, but with continuous laten variable $\mathbf{z}$

$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\Sigma}_{ac})$ - distribution of speaker means

$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mathbf{z}, \boldsymbol{\Sigma}_{wc})$ - within class (channel) variability

- Observations (embeddings) are assumed to be generated as follows:
  - Latent (speaker mean) vector $\mathbf{z}_s$ is generated for each speaker $s$ from gaussian distribution $p(\mathbf{z})$
  - All embeddings of speaker $s$ are generated from Gaussian distribution $p(\mathbf{x}_{si}|\mathbf{z}_s)$