Probabilistic Machine Learning
Lecture 22
Mixture Models & EM

Philipp Hennig

6 July 2021

Eberhard Karls Universität Tübingen
Faculty of Science
Department of Computer Science
Chair for the Methods of Machine Learning

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

k-Means Clustering

Steinhaus, H. (1957). *Sur la division des corps matériels en parties*. Bull. Acad. Polon. Sci. 4 (12): 801–804.

Given $\{x_i\}_{i=1,\dots,n}$

Init Set k means $\{m_k\}$ to random values

Assign each datum x_i to its *nearest mean*. One could denote this by an integer variable

$$k_i = \arg\min_k \|m_k - x_i\|^2$$

or by binary responsibilities

$$r_{ki} = \begin{cases} 1 & \text{if } k_i = k \\ 0 & \text{else} \end{cases}$$

Update set the means to the sample mean of each cluster

$$m_k \leftarrow \frac{1}{R_k} \sum_i^n r_{ki} x_i \qquad \text{where } R_k := \sum_i r_{ki}$$

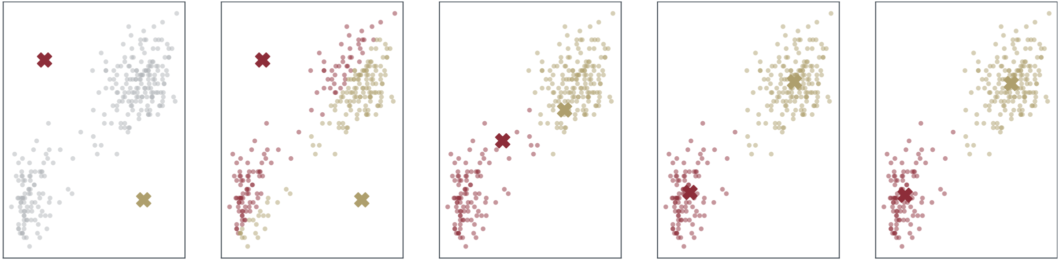
Repeat until the assignments do not change

Hugo Steinhaus
1887–1972

k-Means Clustering

Example on Old Faithful

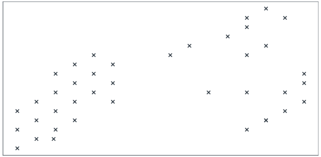
[Figure after in C. Bishop, made by Ann-Kathrin Schalkamp]



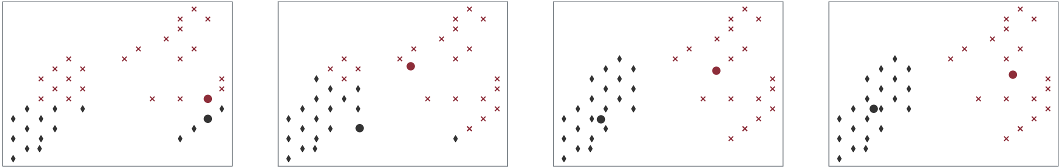
k-means has pathologies

figures after DJC MacKay, ITILA, 2003, recreated by Ann-Kathrin Schalkamp

data from David JC MacKay's book:



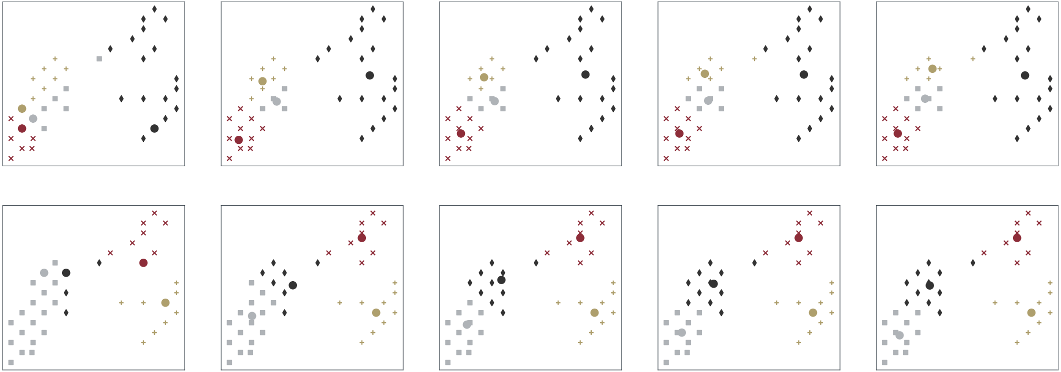
k-means can work well …



k-means has pathologies

figures after DJC MacKay, ITILA, 2003, recreated by Ann-Kathrin Schalkamp

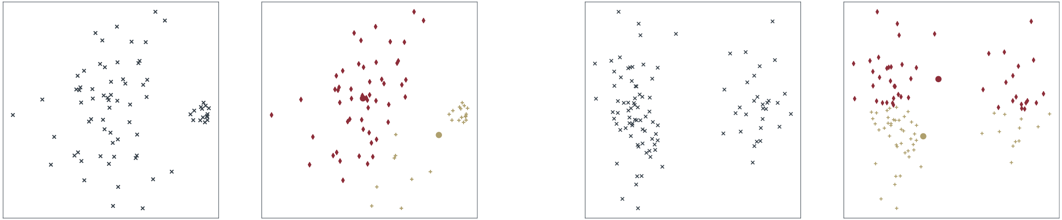
…but it has no way to set k …



k-means has pathologies

figures after DJC MacKay, ITILA, 2003, recreated by Ann-Kathrin Schalkamp

…or to set the *shape* of the clusters!



k-means always converges

for an interesting reason …

Definition (Lyapunov Function)

In the context of iterative algorithms, a *Lyapunov Function* J is a positive function of the algorithm's state variables that decreases in each step of the algorithm.

The existence of a Lyapunov function means that one can think about the algorithm in question as an optimization routine for J. It also guarantees convergence of the algorithm at a *local* (not necessarily global!) minimum of J

Aleksandr M. Lyapunov
(1857—1918)

k-means always converges …

for an interesting reason …


```
1 procedure k-MEANS(x, k)
2     m ← RAND(k)                                   // initialize
3     while not converged do
4         r ← FIND(min(||m − x||^2))                // set responsibilities
5         m ← rx ⊘ r1                               // set means
6     end while
7     return m
8 end procedure
```

Consider $$J(r,m) := \sum_i^n \sum_k^K r_{ik} \|x_i - m_k\|^2$$

- step 4 always decreases J (by definition)
- step 5 always decreases J, because

$$\frac{\partial}{\partial m_k} J(r,m) = -2 \sum_i^n r_{ik}(x_i - m_k) = 0 \quad \Rightarrow \quad m_k = \frac{\sum_i r_{ik} x_i}{\sum_i r_{ik}} \qquad \frac{\partial^2 J(r,m)}{\partial m_k^2} = 2 \sum_i r_{ik} > 0$$

- *k*-means is a simple algorithm that always finds **a** stable clustering
- the resulting clusterings can be unintuitive. They do not capture shape of clusters or their number, and are subject to random fluctuations

a probabilistic interpretation of *k*-means yields clarity and allows fitting all parameters. As a neat side-effect, it leads to a final entry to our toolbox!

From a Loss to a Generative Model

empirical risk minimization is frequently identified with maximum likelihood

$$
\begin{aligned}
(r, m) &= \arg\min_{r,m} \sum_i^n \sum_k^K r_{ik} \|x_i - m_k\|^2 \\
&= \arg\max \sum_i^n \sum_k^K r_{ik} (-{}^1\!/\!_2 \sigma^{-2} \|x_i - m_k\|^2) + \text{const.} \\
&= \arg\max \prod_i^n \sum_k^K r_{ik} \exp\left(-{}^1\!/\!_2 \sigma^{-2} \|x_i - m_k\|^2\right) / Z \\
&= \arg\max \prod_i^n \sum_k^K r_{ik} \mathcal{N}(x_i; m_i, \sigma^2 I) \\
&= \arg\max p(\boldsymbol{x} \mid m, r)
\end{aligned}
$$

k-means maximizes a hard-assignment, isotropic Gaussian mixture model

Gaussian Mixtures

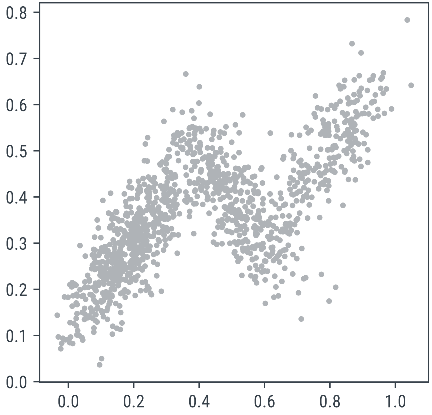
A generative model for *k*-means

[Figure after Bishop, PRML, 2006, by Ann-Kathrin Schalkamp]

$$p(\boldsymbol{x} \mid \pi, \mu, \Sigma) = \prod_i^n \sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)$$

$$\pi_j \in [0, 1],$$

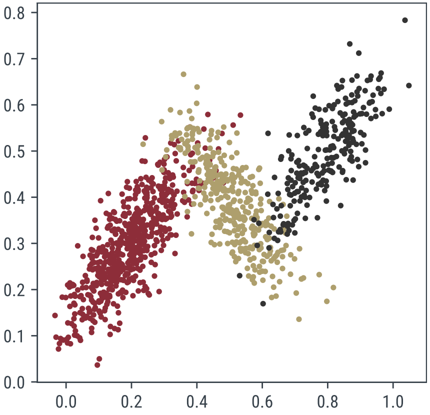
$$\sum_j \pi_j = 1$$



$$p(\boldsymbol{x} \mid \pi, \mu, \Sigma) = \prod_i^n \sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)$$

$$\pi_j \in [0, 1],$$

$$\sum_j \pi_j = 1$$

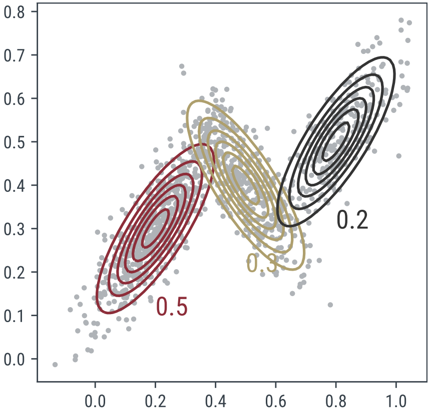


[Figure after Bishop, PRML, 2006, by Ann-Kathrin Schalkamp]

$$p(\mathbf{x} \mid \pi, \mu, \Sigma) = \prod_i^n \sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)$$

$$\pi_j \in [0, 1],$$

$$\sum_j \pi_j = 1$$

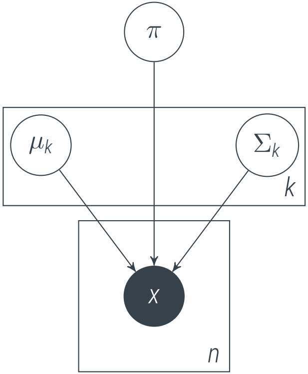


Soft k-means as maximum likelihood

for the Gaussian mixture model

- Given dataset $[x_i]_{i=1,\dots,n}$, want to learn generative model (π, μ, Σ)

$$p(x \mid \pi, \mu, \Sigma) = \prod_i^n \sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \qquad (\star)$$



Soft k-means as maximum likelihood

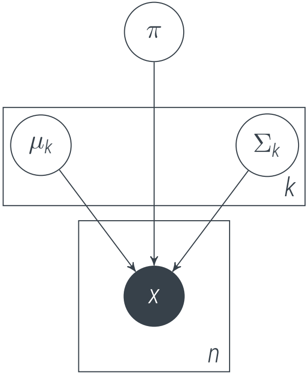
for the Gaussian mixture model

- Given dataset $[x_i]_{i=1,\dots,n}$, want to learn generative model (π, μ, Σ)

$$p(x \mid \pi, \mu, \Sigma) = \prod_i^n \sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \qquad (\star)$$

- Ideally, want Bayesian inference

$$p(\pi, \mu, \Sigma \mid x) = \frac{p(x \mid \pi, \mu, \Sigma) \cdot p(\pi, \mu, \Sigma)}{p(x)}$$



Soft k-means as maximum likelihood

for the Gaussian mixture model

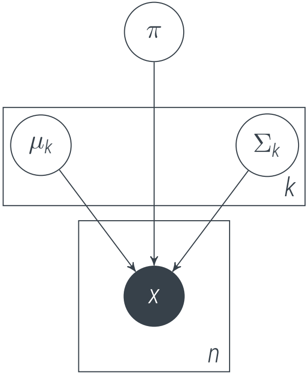
- Given dataset $[x_i]_{i=1,\dots,n}$, want to learn generative model (π, μ, Σ)

$$p(x \mid \pi, \mu, \Sigma) = \prod_i^n \sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \qquad (\star)$$

- Ideally, want Bayesian inference

$$p(\pi, \mu, \Sigma \mid x) = \frac{p(x \mid \pi, \mu, \Sigma) \cdot p(\pi, \mu, \Sigma)}{p(x)}$$

- likelihood is not an exponential family — no obvious conjugate prior



posterior (and likelihood) do not factorize over μ, π, Σ! $\mu \not\perp\!\!\!\perp \pi \mid x$

Soft k-means as maximum likelihood

for the Gaussian mixture model

Let's try to maximize the likelihood ($\star$) for π, μ, Σ (recall $\mathcal{N}(x; \mu, \Sigma) = \frac{e^{-\frac{1}{2}(x-\mu)^\intercal \Sigma^{-1}(x-\mu)}}{(2\pi)^{d/2}|\Sigma|^{1/2}}$)

$$\log p(x \mid \pi, \mu, \Sigma) = \sum_i^n \log \left(\sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \right)$$

To maximize w.r.t. μ set gradient of log likelihood to 0:

$$\nabla_{\mu_j} \log p(x \mid \pi, \mu, \Sigma) = -\sum_i^n \underbrace{\frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}}_{=:r_{ji}} \Sigma_j^{-1}(x_i - \mu_j)$$

$$\nabla_{\mu_j} \log p = 0 \quad \Rightarrow \mu_j = \frac{1}{R_j} \sum_i^n r_{ji} x_i \qquad R_j := \sum_i r_{ji}$$

Soft k-means as maximum likelihood

for the Gaussian mixture model

Let's try to maximize the likelihood ($\star$) for π, μ, Σ (recall $\mathcal{N}(x; \mu, \Sigma) = \frac{e^{-\frac{1}{2}(x-\mu)^\intercal \Sigma^{-1}(x-\mu))}}{(2\pi)^{d/2}|\Sigma|^{1/2}}$)

$$\log p(x \mid \pi, \mu, \Sigma) = \sum_i^n \log \left(\sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \right)$$

To maximize w.r.t. Σ set gradient of log likelihood to 0 (note $\partial|\Sigma|^{-1/2}/\partial\Sigma = -\frac{1}{2}|\Sigma|^{-3/2}|\Sigma|\Sigma^{-1}$ and $\partial(v^\intercal\Sigma^{-1}v)/\partial\Sigma = -\Sigma^{-1}vv^\intercal\Sigma^{-1}$):

$$\nabla_{\Sigma_j} \log p(x \mid \pi, \mu, \Sigma) = -\frac{1}{2} \sum_i^n \underbrace{\frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}}_{=:r_{ji}} \left(\Sigma^{-1}(x_i - \mu_j)(x_i - \mu_j)^\intercal \Sigma^{-1} - \Sigma_j^{-1} \right)$$

$$\nabla_{\Sigma_j} \log p = 0 \quad \Rightarrow \Sigma_j = \frac{1}{R_j} \sum_i^n r_{ji}(x_i - \mu_j)(x_i - \mu_j)^\intercal \qquad R_j := \sum_i r_{ji}$$

Soft k-means as maximum likelihood

for the Gaussian mixture model

Let's try to maximize the likelihood ($\star$) for π, μ, Σ (recall $\mathcal{N}(x; \mu, \Sigma) = \frac{e^{-\frac{1}{2}(x-\mu)^\intercal \Sigma^{-1}(x-\mu))}}{(2\pi)^{d/2}|\Sigma|^{1/2}}$)

$$\log p(x \mid \pi, \mu, \Sigma) = \sum_i^n \log \left(\sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \right)$$

To maximize w.r.t. π, enforce $\sum_j \pi_j = 1$ by introducing Lagrange multiplier λ and optimize

$$\nabla_{\pi_j} \log p(x \mid \pi, \mu, \Sigma) + \lambda \left(\sum_j \pi_j - 1 \right) = \sum_i^n \frac{\mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)} + \lambda$$

$$0 = \sum_i^n \pi_j \frac{\mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)} + \lambda \pi_j = \sum_i^n r_{ij} + \lambda \pi_j$$

$$\sum_j \pi_j = 1 \Rightarrow \lambda = -N \qquad \Rightarrow \qquad \pi_j = \frac{R_j}{n}$$

The EM Algorithm (for Gaussian mixtures)

Refinement of soft *k*-means and *k*-means

If we know the responsibilities r_{ij}, we can optimize μ, Σ, π analytically. And if we know μ, π, we can set r_{ij}! Thus

1. initialize μ, π (e.g. random μ, uniform π)
2. Set

$$r_{ij} = \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'}^{k} \pi_{j'} \mathcal{N}(x_i; \mu_{j'}, \Sigma_{j'})}$$

3. Set

$$R_j = \sum_i r_{ji} \qquad \mu_j = \frac{1}{R_j} \sum_i^n r_{ij} x_i \qquad \Sigma_j = \frac{1}{R_j} \sum_i^n r_{ij} (x_i - \mu_j)(x_i - \mu_j)^\intercal \qquad \pi_j = \frac{R_j}{n}$$

- Note that π is essentially given through r_{ij}, thus can be incorporated into the first step

The connection to (soft) k-means

Refinement of soft k-means and k-means with cluster probabilities

Set $\Sigma_j = \beta^{-1} I$ for all $j = 1, \dots, k$

1. initialize μ, π (e.g. random μ, uniform π)
2. Set

$$r_{ij} = \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'}^{k} \pi_{j'} \mathcal{N}(x_i; \mu_{j'}, \Sigma_{j'})} = \frac{R_j \exp(-\beta \|x_i - m_j\|^2)}{\sum_{j'} R_{j'} \exp(-\beta \|x_i - m_{j'}\|^2)}$$

3. Set

$$R_j = \sum_i r_{ji} \qquad \mu_j = \frac{1}{R_j} \sum_i^n r_{ij} x_i \qquad \left(\Sigma_j = \frac{1}{R_j} \sum_i^n r_{ij} (x_i - \mu_j)(x_i - \mu_j)^\intercal \qquad \pi_j = \frac{R_j}{n} \right)$$

the EM algorithm is a refinement of soft k-means

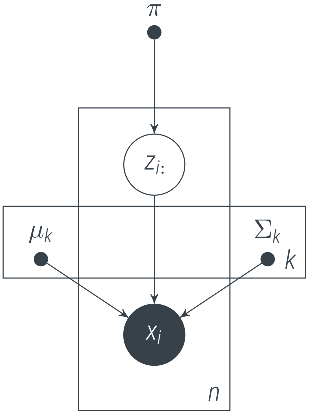
- For $\beta \to \infty$, get back k-means
- What is r_{ij}?

- consider binary $z_{ij} \in \{0; 1\}$ with $\sum_j z_{ij} = 1$ ("one-hot")
- what is $p(x, z)$? Let's write it as $p(x, z) = p(x \mid z)p(z)$ with

$$p(z_{ij} = 1) = \pi_j \qquad \Rightarrow p(z) = \prod_i^n \prod_j^k \pi_j^{z_{ij}}$$

$$p(x_i \mid z_j = 1) = \mathcal{N}(x_i; \mu_j, \Sigma_j) \qquad \Rightarrow p(x_i \mid z_{i:}) = \prod_j^k \mathcal{N}(x \mid \mu_j, \Sigma_j)^{z_{ij}}$$

$$p(x_i) = \sum_j p(z = j)p(x_i \mid z = j) = \sum_j^k \pi_j \mathcal{N}(x; \mu_j, \Sigma_j)$$



Joint Generative Model

For the Gaussian Mixture

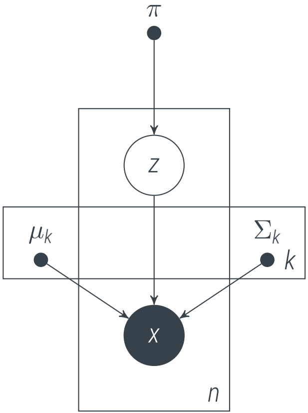
$$p(x, z \mid \pi, \mu, \Sigma) = \prod_i^n \prod_j^k \pi_j^{z_{ij}} \mathcal{N}(x_i; \mu_j, \Sigma_j)^{z_{ij}}$$

$$p(z_{ij} = 1 \mid x_i, \mu, \Sigma) = \frac{p(z_{ij} = 1) p(x_i \mid z_{ij} = 1, \mu_j, \Sigma_j)}{\sum_{j'}^k p(z_{ij'} = 1) p(x_i \mid z_{ij'} = 1, \mu_j, \Sigma_j)}$$

$$= \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}$$

$$= r_{ij}$$

r_{ij} is the marginal posterior probability (**[E]xpectation**) for $z_{ij} = 1$!



Given μ, Σ, have a simple distribution for z. And, given z, μ, Σ show up in a tractable form.

The Expectation Maximization Algorithm

Refinement of soft k-means and k-means with cluster probabilities

Set $\Sigma_j = \beta^{-1} I$ for all $j = 1, \dots, k$

1. initialize μ, π (e.g. random μ, uniform π)
2. Compute **EXPECTED** value of z:

$$r_{ij} = \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'}^{k} \pi_{j'} \mathcal{N}(x_i; \mu_{j'}, \Sigma_{j'})} = \frac{R_j \exp(-\beta \|x_i - m_j\|^2)}{\sum_{j'} R_{j'} \exp(-\beta \|x_i - m_{j'}\|^2)}$$

3. **MAXIMIZE** Likelihood

$$R_j = \sum_i r_{ji} \qquad \mu_j = \frac{1}{R_j} \sum_i^n r_{ij} x_i \qquad \left(\Sigma_j = \frac{1}{R_j} \sum_i^n r_{ij} (x_i - \mu_j)(x_i - \mu_j)^\intercal \qquad \pi_j = \frac{R_j}{n} \right)$$

the EM algorithm is an *iterative maximum likelihood* algorithm.

Taking the easy way out

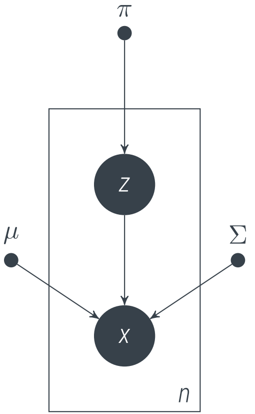
Just pretend you know that variable that causes trouble

$$p(x, z \mid \pi, \mu, \Sigma) = \prod_i^n \prod_j^k \pi_j^{z_{ij}} \mathcal{N}(x_i; \mu_j, \Sigma_j)^{z_{ij}}$$

$$p(x \mid z, \pi, \mu, \Sigma) = \prod_i^n \pi_{k_i} \mathcal{N}(x_i; \mu_{k_i}, \Sigma_{k_i})$$

$$\pi_j \leftarrow \frac{N_j}{N} \qquad N_j = \sum_i z_{ij}$$

$$\mu_j \leftarrow \frac{1}{N_j} \sum_i z_{ij} x_i \qquad \Sigma_j \leftarrow \frac{1}{N_j} \sum_i z_{ij}(x_i - \mu_j)(x_i - \mu_j)^\intercal$$



But we didn't have z! So, for EM, we replaced it with its expectation!

The Toolbox

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Framework:

$$\int p(x_1, x_2)\, dx_2 = p(x_1) \qquad p(x_1, x_2) = p(x_1 \mid x_2)p(x_2) \qquad p(x \mid y) = \frac{p(y \mid x)p(x)}{p(y)}$$

Modelling:

- graphical models
- Gaussian distributions
- (deep) learnt representations
- Kernels
- Markov Chains
- Exponential Families / Conjugate Priors
- Factor Graphs & Message Passing

Computation:

- Monte Carlo
- Linear algebra / Gaussian inference
- maximum likelihood / MAP
- Laplace approximations
- EM
- Variational approximations

Generic EM Algorithm

Maximize **expected** log likelihoods

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Setting:

- Want to find *maximum likelihood* (or MAP) estimate for a model involving a **latent** variable

$$\theta_* = \arg\max_\theta \left[\log p(x \mid \theta)\right] = \arg\max_\theta \left[\log\left(\sum_z p(x, z \mid \theta)\right)\right]$$

- Assume that the summation inside the log makes analytic optimization intractable
- but that optimization would be analytic if z were known (i.e. if there were only one term in the sum)

Idea: Initialize θ_0, then iterate between

1. Compute $p(z \mid x, \theta_{\text{old}})$
2. Set θ_{new} to the **Maximum** of the **Expectation** of the *complete-data* log likelihood:

$$\theta_{\text{new}} = \arg\max_\theta \sum_z p(z \mid x, \theta_{\text{old}}) \log p(\underbrace{x, z}_{!} \mid \theta) = \arg\max_\theta \mathbb{E}_{p(z \mid x, \theta_{\text{old}})}\left[\log p(x, z \mid \theta)\right]$$

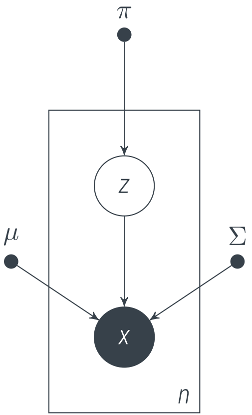
3. Check for convergence of either the log likelihood, or θ.

EM for Gaussian Mixtures

re-written in generic form

- Want to maximize, as function of $\theta := (\pi_j, \mu_j, \Sigma_j)_{j=1,\dots,k}$

$$\log p(x \mid \pi, \mu, \Sigma) = \sum_i \log \left(\sum_j \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \right)$$



EM for Gaussian Mixtures

re-written in generic form

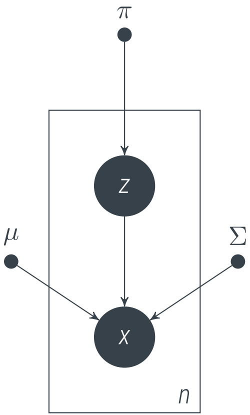
- Want to maximize, as function of $\theta := (\pi_j, \mu_j, \Sigma_j)_{j=1,\dots,k}$

$$\log p(x \mid \pi, \mu, \Sigma) = \sum_i \log \left(\sum_j \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \right)$$

- Instead, maximizing the “complete data” likelihood is easier:

$$\log p(x, z \mid \pi, \mu, \Sigma) = \log \prod_i^n \prod_j^k \pi_j^{z_{ij}} \mathcal{N}(x_i; \mu_j, \Sigma_j)^{z_{ij}}$$

$$= \sum_i \sum_j z_{nk} \underbrace{(\log \pi_j + \log \mathcal{N}(x_i; \mu_j, \Sigma_j)))}_{\text{easy to optimize (exponential families!)}}$$



EM for Gaussian Mixtures

re-written in generic form

1. Compute $p(z \mid x, \theta)$:

$$p(z_{ij} = 1 \mid x_i, \mu, \Sigma) = \frac{p(z_{ij} = 1) p(x_i \mid z_{ij} = 1)}{\sum_{j'}^{k} p(z_{ij'} = 1) p(x_i \mid z_{ij'} = 1)} = \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)} =: r_{ij}$$

2. Maximize

$$\mathbb{E}_{p(z|x,\theta)} (\log p(x, z \mid \theta)) = \sum_i \sum_j r_{ij} \left(\log \pi_j + \log \mathcal{N}(x_i; \mu_j, \Sigma_j)\right)$$

(see earlier slides on how to solve this, much easier problem)

The EM algorithm

Instead of trying to maximize

$$\log p(x \mid \theta) = \log \sum_z p(x, z \mid \theta) = \log \sum_z p(z \mid x, \theta) p(x \mid \theta),$$

instead maximize

$$\mathbb{E}_z \log p(x, z \mid \theta) = \sum_z p(z \mid x, \theta) \log p(x, z \mid \theta),$$

then re-compute $p(z \mid x, \theta)$, and repeat.