<table>
<thead>
<tr>
<th>#</th>
<th>content</th>
<th>Ex</th>
<th>#</th>
<th>content</th>
<th>Ex</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>1</td>
<td>15</td>
<td>Exponential Families</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>Reasoning under Uncertainty</td>
<td>16</td>
<td>16</td>
<td>Graphical Models</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Continuous Variables</td>
<td>2</td>
<td>17</td>
<td>Factor Graphs</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>Monte Carlo</td>
<td>18</td>
<td>18</td>
<td>The Sum-Product Algorithm</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Markov Chain Monte Carlo</td>
<td>3</td>
<td>19</td>
<td>Example: Modelling Topics</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>Gaussian Distributions</td>
<td>2</td>
<td>20</td>
<td>Latent Dirichlet Allocation</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Parametric Regression</td>
<td>4</td>
<td>21</td>
<td>k-mneans</td>
<td>11</td>
</tr>
<tr>
<td>8</td>
<td>Learning Representations</td>
<td>22</td>
<td>22</td>
<td>Mixture Models</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Gaussian Processes</td>
<td>5</td>
<td>23</td>
<td>Expectation Maximization (EM)</td>
<td>12</td>
</tr>
<tr>
<td>10</td>
<td>Understanding Kernels</td>
<td>24</td>
<td>24</td>
<td>Variational Inference</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Gauss-Markov Models</td>
<td>6</td>
<td>25</td>
<td>Example: Collapsed Inference</td>
<td>13</td>
</tr>
<tr>
<td>12</td>
<td>An Example for GP Regression</td>
<td>7</td>
<td>26</td>
<td>Example: EM for Kernel Topic Models</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>GP Classification</td>
<td>27</td>
<td>27</td>
<td>Making Decisions</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Generalized Linear Models</td>
<td></td>
<td>28</td>
<td>Revision</td>
<td></td>
</tr>
</tbody>
</table>
Given \( \{x_i\}_{i=1,...,n} \)

- **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

  \[
  m_k \leftarrow \frac{1}{R_k} \sum_{i} r_{ki} x_i \quad \text{where } R_k := \sum_{i} r_{ki}
  \]

- **Repeat** until the assignments do not change.
k-Means Clustering

Example on Old Faithful

[Figure after in C. Bishop, made by Ann-Kathrin Schalkamp]
$k$-means has pathologies

data from David JC MacKay's book:

$k$-means can work well...
k-means has pathologies

...but it has no way to set $k$ ...

figures after DJC MacKay, ITILA, 2003, recreated by Ann-Kathrin Schalkamp
k-means has pathologies

...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 ...

Consider $J(r, m) := \sum_{i=1}^{n} \sum_{k=1}^{K} r_{ik} \|x_i - m_k\|^2$

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

$$\frac{\partial}{\partial m_k} J(r, m) = -2 \sum_{i}^{n} r_{ik} (x_i - m_k) = 0 \Rightarrow m_k = \frac{\sum_{i} r_{ik} x_i}{\sum_{i} r_{ik}} \quad \frac{\partial^2 J(r, m)}{\partial m^2_k} = 2 \sum_{i} r_{ik} > 0$$
- *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!
(r, m) = \arg \min_{r,m} \sum_i \sum_k r_{ik} ||x_i - m_k||^2

= \arg \max \sum_i \sum_k r_{ik} (\frac{-1}{2} \sigma^{-2} ||x_i - m_k||^2) + \text{const.}

= \arg \max \prod_i \sum_k r_{ik} \exp \left( \frac{-1}{2} \sigma^{-2} ||x_i - m_k||^2 \right) / Z

= \arg \max \prod_i \sum_k r_{ik} \mathcal{N}(x_i; m_i, \sigma^2 I)

= \arg \max p(x \mid m, r)

\textit{k-means maximizes a hard-assignment, isotropic Gaussian mixture model}
Gaussian Mixtures
A generative model for $k$-means

\[ p(x \mid \pi, \mu, \Sigma) = \prod_{i} \sum_{j} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \]

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

\[ \sum_{j} \pi_j = 1 \]
Gaussian Mixtures
A generative model for k-means

\[ p(x \mid \pi, \mu, \Sigma) = \prod_{i} \sum_{j} \pi_{j} \mathcal{N}(x_{i}; \mu_{j}, \Sigma_{j}) \]

\[ \pi_{j} \in [0, 1], \quad \sum_{j} \pi_{j} = 1 \]
Gaussian Mixtures

A generative model for $k$-means

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

$$p(x \mid \pi, \mu, \Sigma) = \prod_{i}^{n} \sum_{j}^{k} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)$$

$$\pi_j \in [0, 1], \quad \sum_{j} \pi_j = 1$$
Soft $k$-means as maximum likelihood for the Gaussian mixture model

Given dataset $[x_i]_{i=1,...,n}$, want to learn generative model $(\pi, \mu, \Sigma)$

\[
p(x \mid \pi, \mu, \Sigma) = \prod_{i}^{n} \sum_{j}^{k} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)
\]  

(\star)
Soft \( k \)-means as maximum likelihood for the Gaussian mixture model

Given dataset \([x_i]_{i=1,...,n}\), want to learn generative model \((\pi, \mu, \Sigma)\)

\[
p(x \mid \pi, \mu, \Sigma) = \prod_{i}^{n} \sum_{j}^{k} \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)
\]  

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 $[x_i]_{i=1,...,n}$, want to learn generative model $(\pi, \mu, \Sigma)$

\[ p(x \mid \pi, \mu, \Sigma) = \prod_i \sum_j \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \]  

- 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 $\mu, \pi, \Sigma$! $\mu \not\perp \pi \mid x$
Soft $k$-means as maximum likelihood
for the Gaussian mixture model

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

$$
\log p(x \mid \pi, \mu, \Sigma) = \sum_{i}^{n} \log \left( \sum_{j}^{k} \pi_{j} \mathcal{N}(x_{i}; \mu_{j}, \Sigma_{j}) \right)
$$

To maximize w.r.t. $\mu$ set gradient of log likelihood to 0:

$$
\nabla_{\mu_{j}} \log p(x \mid \pi, \mu, \Sigma) = - \sum_{i}^{n} \left( \frac{\pi_{j} \mathcal{N}(x_{i}; \mu_{j}, \Sigma_{j})}{\sum_{j'} \pi_{j'} \mathcal{N}(x_{i}; \mu_{j'}, \Sigma_{j'})} \right) \Sigma_{j}^{-1} (x_{i} - \mu_{j})
$$

$$
\nabla_{\mu_{j}} \log p = 0 \quad \Rightarrow \quad \mu_{j} = \frac{1}{R_{j}} \sum_{i}^{n} r_{ji} x_{i} \quad R_{j} := \sum_{i} r_{ji}
$$
Let's try to maximize the likelihood (⋆) for \( \pi, \mu, \Sigma \) (recall \( \mathcal{N}(x; \mu, \Sigma) = \frac{e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)}}{(2\pi)^{d/2} |\Sigma|^{1/2}} \))

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

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

\[
\nabla_{\Sigma_j} \log p(x \mid \pi, \mu, \Sigma) = -\frac{1}{2} \sum_i^n \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_j^k \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)} \left( \Sigma_j^{-1}(x_i - \mu_j)(x_i - \mu_j)^\top \Sigma_j^{-1} - \Sigma_j^{-1} \right)
\]

\[
= r_{ji}
\]

\[
\nabla_{\Sigma_j} \log p = 0 \quad \Rightarrow \quad \Sigma_j = \frac{1}{R_j} \sum_i^n r_{ji}(x_i - \mu_j)(x_i - \mu_j)^\top \quad R_j := \sum_i r_{ji}
\]
Soft $k$-means as maximum likelihood for the Gaussian mixture model

Let’s try to maximize the likelihood ($\star$) for $\pi, \mu, \Sigma$ (recall $\mathcal{N}(x; \mu, \Sigma) = \frac{e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}}{(2\pi)^{d/2} |\Sigma|^{1/2}}$)

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

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

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

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

$$\sum \pi_j = 1 \Rightarrow \lambda = -N \Rightarrow \pi_j = \frac{R_j}{n}$$
If we know the responsibilities $r_{ij}$, we can optimize $\mu, \Sigma, \pi$ analytically. And if we know $\mu, \pi$, we can set $r_{ij}$. Thus

1. initialize $\mu, \pi$ (e.g. random $\mu$, uniform $\pi$)
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
   
   $R_j = \sum_i r_{ji}$  \hspace{1cm} $\mu_j = \frac{1}{R_j} \sum_i r_{ij} x_i$  \hspace{1cm} $\Sigma_j = \frac{1}{R_j} \sum_i r_{ij} (x_i - \mu_j)(x_i - \mu_j)^\top$  \hspace{1cm} $\pi_j = \frac{R_j}{n}$


Note that $\pi$ 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 $j = 1, \ldots, k$

1. initialize $\mu, \pi$ (e.g. random $\mu$, uniform $\pi$)

2. Set

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

3. Set

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

the EM algorithm is a refinement of soft $k$-means

- For $\beta \to \infty$, get back $k$-means
- What is $r_{ij}$?
Gaussian Mixtures Revisited
Introducing a Latent Variable Simplifies Things

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

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

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

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

For the Gaussian Mixture

\[
p(x, z \mid \pi, \mu, \Sigma) = \prod_i \prod_j \pi_i \mathcal{N}(x_i; \mu_j, \Sigma_j)^{z_{ij}}
\]

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

\[
= \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_{j'} \mathcal{N}(x_i; \mu_{j'}, \Sigma_{j'})}
\]

\[
= r_{ij}
\]

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

Given \(\mu, \Sigma\), have a simple distribution for \(z\). And, given \(z, \mu, \Sigma\) 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 $j = 1, \ldots, k$

1. initialize $\mu, \pi$ (e.g. random $\mu$, uniform $\pi$)
2. Compute EXPECTED value of $z$:

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

3. MAXIMIZE Likelihood

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

the EM algorithm is an iterative maximum likelihood algorithm.
Taking the easy way out

Just pretend you know that variable that causes trouble

\[
p(x, z \mid \pi, \mu, \Sigma) = \prod_{i}^{n} \prod_{j}^{k} \pi_j^{z_{ij}} \mathcal{N}(x_i; \mu_j, \Sigma_j)^{z_{ij}}
\]

\[
p(x \mid z, \pi, \mu, \Sigma) = \prod_{i}^{n} \pi_k^{z_{ij}} \mathcal{N}(x_i; \mu_k, \Sigma_k)
\]

\[
\pi_j \leftarrow \frac{N_j}{N} \quad N_j = \sum_{i} Z_{ij}
\]

\[
\mu_j \leftarrow \frac{1}{N_j} \sum_{i} Z_{ij} x_i \quad \Sigma_j \leftarrow \frac{1}{N_j} \sum_{i} Z_{ij} (x_i - \mu_j)(x_i - \mu_j)^T
\]

But we didn’t have \( z \)! So, for EM, we replaced it with its expectation!
The Toolbox

Framework:

\[
\int p(x_1, x_2) \, dx_2 = p(x_1) \quad p(x_1, x_2) = p(x_1 | x_2)p(x_2) \quad p(x | y) = \frac{p(y | x)p(x)}{p(y)}
\]

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

Setting:

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

\[
\theta^* = \arg \max_\theta [\log p(x | \theta)] = \arg \max_\theta \left[ \log \left( \sum_z p(x, z | \theta) \right) \right]
\]

- 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 \( \theta_0 \), then iterate between

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

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

3. Check for convergence of either the log likelihood, or \( \theta \).
Want to maximize, as function of \( \theta := (\pi_j, \mu_j, \Sigma_j)_{j=1,\ldots,k} \)

\[
\log p(x \mid \pi, \mu, \Sigma) = \sum_i \log \left( \sum_j \pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j) \right)
\]
EM for Gaussian Mixtures

re-written in generic form

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

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

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

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

\[
= \sum_i \sum_j z_{nk} \left( \log \pi_j + \log \mathcal{N}(x_i; \mu_j, \Sigma_j) \right)
\]

easy to optimize (exponential families!)
EM for Gaussian Mixtures

re-written in generic form

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

\[
p(z_{ij} = 1 \mid x_i, \mu, \Sigma) = \frac{p(z_{ij} = 1)p(x_i \mid z_{ij} = 1)}{\sum_{j'} p(z_{ij'} = 1)p(x_i \mid z_{ij'} = 1)} = \frac{\pi_j \mathcal{N}(x_i; \mu_j, \Sigma_j)}{\sum_{j'} \pi_{j'} \mathcal{N}(x_i; \mu_{j'}, \Sigma_{j'})} =: r_{ij}
\]

2. Maximize

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

(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.