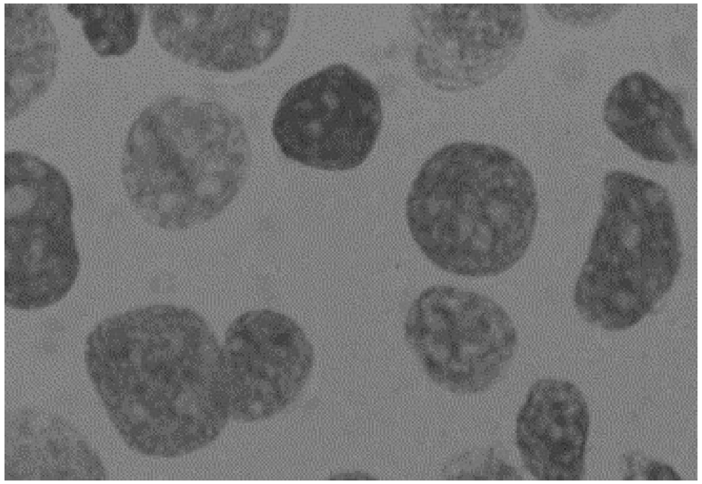
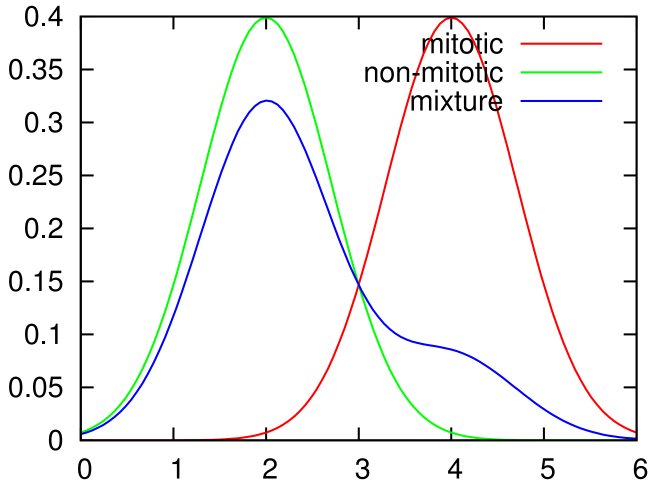
Expectation Maximisation Algorithm

Boris Flach

A. Examples

Example 1. Fraction of mitotic cells





- stain DNA $\Rightarrow$ segment nuclei $\Rightarrow$ $x \in \mathbb{R}$ total stain of a nucleus
- two classes $k \in \{1, 2\}$ non-mitotic, mitotic

$$p(x) = p(k=1)\,\mathcal{N}(x;\, \mu, \sigma) + p(k=2)\,\mathcal{N}(x;\, 2\mu, \sigma)$$

where μ, σ known.

Training data: $\mathcal{T} = \{x_1, \ldots, x_\ell\}$ total stain for a sample of nuclei

Task: estimate $p(k)$, $k = 1, 2$.

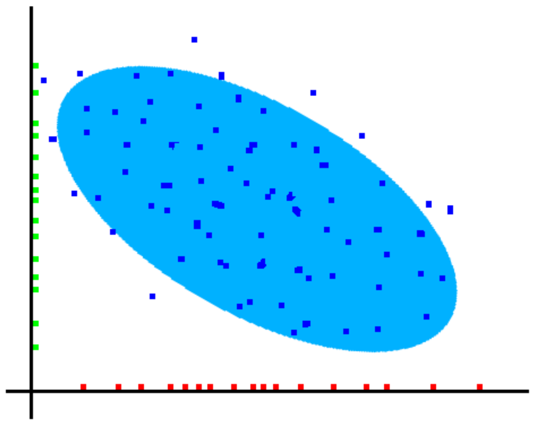
A. Examples

Example 2. Bivariate normal distribution

$$\boldsymbol{r} = (x, y) \in \mathbb{R}^2, \quad p(\boldsymbol{r}) = \mathcal{N}(\boldsymbol{r};\, \boldsymbol{\mu}, S)$$

mean $\boldsymbol{\mu}$ and covariance matrix S are unknown.

Training data:



- $\mathcal{T}_1 = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_k\}$ sample with complete information,
- $\mathcal{T}_2 = \{x_1, \ldots, x_\ell\}$ sample with y coordinate missing,
- $\mathcal{T}_3 = \{y_1, \ldots, y_m\}$ sample with x coordinate missing

Task: estimate $\boldsymbol{\mu}$, S.

A. Examples

Example 3. Segmentation

- segment images into K segments. Appearance for segment $k \in K$: $p(\boldsymbol{x} \mid k)$, where $\boldsymbol{x} \in \mathbb{R}^3$ denotes colour.
- assumption for the colour distribution of a segment – mixture of Gaussians

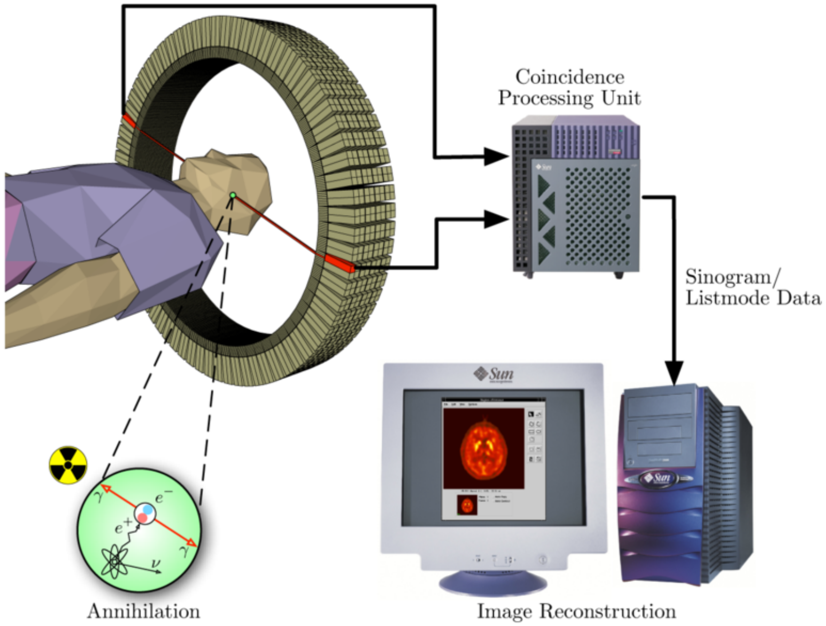
$$p(\boldsymbol{x}) = \sum_{m=1}^{M} \pi_m \cdot \mathcal{N}(\boldsymbol{x};\, \boldsymbol{\mu}_m, S_m)$$

Training data: $\mathcal{T} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell\}$ sample of colour values for the segment

Task: estimate π_m, $\boldsymbol{\mu}_m$, S_m, $m = 1, \ldots, M$.

A. Examples

Example 4. Positron Emission Tomography



- Density of the tracer $\rho(\boldsymbol{r})$, $\boldsymbol{r} \in D \subset \mathbb{Z}^3$
- Number of positron emissions $n(\boldsymbol{r})$ is a random variable $p\big(n(\boldsymbol{r}) \mid \rho(\boldsymbol{r})\big)$
- measurement – coincident photon pairs $\Rightarrow$ line

Data: $\mathcal{T} = \{l_1, \ldots, l_m\}$ sample of lines

Task: estimate $\rho(\boldsymbol{r})$, $\boldsymbol{r} \in D$.

B. Formal task

All examples have in common:

- model: joint p.d. for a set of random variables, known up to parameters
- training data: i.i.d. sample for a subset of the variables (or a function of them),
- task: estimate the parameters of the p.d.

Model:

- $x \in \mathcal{X}$ features, $k \in K$ state of the object, joint p.d. $p(x,k) = p(x \mid k;\, \theta_k) \cdot p(k)$
- $p(x \mid k;\, \theta_k) \in \mathcal{P}_\Theta$ model class
- unknown parameters $\{\theta_k, p(k) \mid k \in K\} = m \in \mathcal{M}$

Training data: $\mathcal{T} = \{x_1, \ldots, x_\ell\}$

Maximum Likelihood Estimate:

$$m^* = \arg\max_{m \in \mathcal{M}} \prod_{j=1}^{\ell} \sum_{k \in K} p(x_j, k;\, m) = \arg\max_{m \in \mathcal{M}} \sum_{j=1}^{\ell} \log \sum_{k \in K} p(x_j \mid k;\, \theta_k) \cdot p(k)$$

Remark: learning with complete data is much easier i.e. $\mathcal{T} = \big\{(x_1, k_1), \ldots, (x_\ell, k_\ell)\big\}$

C. The Expectation Maximisation algorithm

The EM algorithm is iterative. The model estimate $m^{(t)}$ is improved in each iteration $t = 1, 2, \ldots$ by using the training data $\mathcal{T}$ and the previous estimate $m^{(t-1)}$.

Init: choose a model $m^{(0)}$

Iterate:

E-step compute

$$\beta_j^{(t)}(k) = p(k \mid x_j;\, m^{(t-1)}),\ \ \forall k \in K,\ \ \forall x_j \in \mathcal{T}$$

M-step re-estimate the model

$$p^{(t)}(k) = \frac{1}{\ell} \sum_{j=1}^{\ell} \beta_j^{(t)}(k)$$

$$\theta_k^{(t)} = \arg\max_{\theta} \sum_{j=1}^{\ell} \beta_j^{(t)}(k)\, \log p(x_j \mid k;\, \theta),\ \ \forall k \in K$$

Stop: if $\|\beta^{(t)} - \beta^{(t-1)}\| < \epsilon$

D. Why it works

Lemma Let $\beta_i > 0$, $i = 1, \dots, n$ be real numbers s.t. $\sum_i \beta_i = 1$. The optimisation task

$$\sum_{i=1}^{n} \beta_i \log x_i \to \max_x$$

$$\text{s.t. } x \in \mathbb{R}^n_+, \quad \sum_{i=1}^{n} x_i = 1$$

has a unique maximiser $x_i^* = \beta_i$, $i = 1, \dots, n$. ■

Consider the objective function of the ML-estimate. Let $\beta_j(k) \geqslant 0$ be arbitrary real numbers s.t. $\sum_{k \in K} \beta_j(k) = 1$, $\forall j = 1, 2, \dots, \ell$.

$$L(m) = \sum_{j=1}^{\ell} \log \sum_{k \in K} p(x_j \mid k;\, \theta_k) \cdot p(k) = \sum_{j,k} \beta_j(k)\, \log \sum_{k' \in K} p(x_j \mid k';\, \theta_{k'}) \cdot p(k')$$

$$= \sum_{j,k} \beta_j(k)\, \log \big[p(x_j \mid k;\, \theta_k) \cdot p(k) \big] - \sum_{j,k} \beta_j(k)\, \log \frac{p(x_j \mid k;\, \theta_k) \cdot p(k)}{\sum_{k' \in K} p(x_j \mid k';\, \theta_{k'}) \cdot p(k')}$$

D. Why it works

The expression under the logarithm in the second term is the class-posterior, hence, we get

$$L(m) = \sum_{j,k} \beta_j(k) \log\bigl[p(x_j \mid k;\, \theta_k) \cdot p(k)\bigr] - \sum_{j,k} \beta_j(k) \log p(k \mid x_j;\, m).$$

If we choose $\beta_j^{(t)}(k) = p(k \mid x_j;\, m^{(t)})$ $\Rightarrow$ any change of m will decrease the second term $\Rightarrow$ we don't need to care about this term $\Rightarrow$ choose the new estimate of $m^{(t+1)}$ so as to maximise the first term of $L(m)$.

Positive:

- E-step and M-step are computationally simpler than direct maximisation of $L(m)$.
- the sequence $L(m^{(t)})$ is increasing, the sequence $\beta^{(t)}$ is convergent.

Negative:

- The algorithm converges to local maxima of $L(m)$.