

## LECTURE 9:

## THE EM ALGORITHM

February 8, 2006

- Iterative algorithm with two linked steps:
  - **E-step:** fill in values of  $\hat{\mathbf{z}}^t$  using  $p(\mathbf{z}|\mathbf{x}, \theta^t)$ .
  - **M-step:** update parameters using  $\theta^{t+1} \leftarrow \operatorname{argmax} \ell(\theta; \mathbf{x}, \hat{\mathbf{z}}^t)$ .
- E-step involves inference, which we need to do at runtime anyway. M-step is no harder than in fully observed case.
- We will prove that this procedure monotonically improves  $\ell$  (or leaves it unchanged). Thus it always converges to a local optimum of the likelihood (as any optimizer should).
- Note: EM is an optimization strategy for objective functions that can be interpreted as likelihoods in the presence of missing data.
- EM is *not* a cost function such as “maximum-likelihood”. EM is *not* a model such as “mixture-of-Gaussians”.

- With latent variables, the probability contains a sum, so the log likelihood has all parameters coupled together:

$$\ell(\theta; \mathcal{D}) = \log \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}|\theta) = \log \sum_{\mathbf{z}} p(\mathbf{z}|\theta_z) p(\mathbf{x}|\mathbf{z}, \theta_x)$$

(we can also consider continuous  $\mathbf{z}$  and replace  $\sum$  with  $\int$ )

- If the latent variables were observed, parameters would decouple again and learning would be easy:

$$\ell(\theta; \mathcal{D}) = \log p(\mathbf{x}, \mathbf{z}|\theta) = \log p(\mathbf{z}|\theta_z) + \log p(\mathbf{x}|\mathbf{z}, \theta_x)$$

- One idea: ignore this fact, compute  $\partial \ell / \partial \theta$ , and do learning with a smart optimizer like conjugate gradient.
- Another idea: what if we use our current parameters to *guess* the values of the latent variables, and then do fully-observed learning? This back-and-forth trick might make optimization easier.

- Observed variables  $\mathbf{x}$ , latent variables  $\mathbf{z}$ , parameters  $\theta$ :

$$\ell_c(\theta; \mathbf{x}, \mathbf{z}) = \log p(\mathbf{x}, \mathbf{z}|\theta)$$

is the *complete log likelihood*.

- Usually optimizing  $\ell_c(\theta)$  given both  $\mathbf{z}$  and  $\mathbf{x}$  is straightforward. (e.g. class conditional Gaussian fitting, linear regression)
- With  $\mathbf{z}$  unobserved, we need the log of a marginal probability:

$$\ell(\theta; \mathbf{x}) = \log p(\mathbf{x}|\theta) = \log \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}|\theta)$$

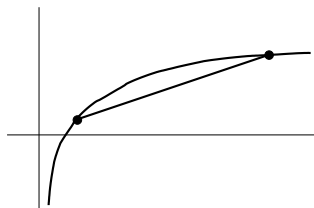
which is the *incomplete log likelihood*.

- For any distribution  $q(\mathbf{z})$  define *expected complete log likelihood*:

$$\ell_q(\theta; \mathbf{x}) = \langle \ell_c(\theta; \mathbf{x}, \mathbf{z}) \rangle_q \equiv \sum_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log p(\mathbf{x}, \mathbf{z}|\theta)$$

- Amazing fact:  $\ell(\theta) \geq \ell_q(\theta) + \mathcal{H}(q)$  because of concavity of log:

$$\begin{aligned} \ell(\theta; \mathbf{x}) &= \log p(\mathbf{x}|\theta) \\ &= \log \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}|\theta) \\ &= \log \sum_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z}|\mathbf{x})} \\ &\geq \sum_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z}|\mathbf{x})} \end{aligned}$$



- Where the inequality is called *Jensen's inequality*.  
(It is only true for distributions:  $\sum q(\mathbf{z}) = 1$ ;  $q(\mathbf{z}) > 0$ .)

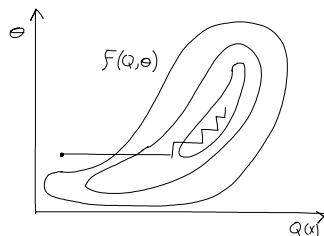
- For fixed data  $\mathbf{x}$ , define a functional called the *free energy*:

$$F(q, \theta) \equiv \sum_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z}|\mathbf{x})} \leq \ell(\theta)$$

- The EM algorithm is coordinate-ascent on  $F$ :

**E-step:**  $q^{t+1} = \operatorname{argmax}_q F(q, \theta^t)$

**M-step:**  $\theta^{t+1} = \operatorname{argmax}_\theta F(q^{t+1}, \theta^t)$



- Note that the free energy breaks into two terms:

$$\begin{aligned} F(q, \theta) &= \sum_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log \frac{p(\mathbf{x}, \mathbf{z}|\theta)}{q(\mathbf{z}|\mathbf{x})} \\ &= \sum_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log p(\mathbf{x}, \mathbf{z}|\theta) - \sum_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log q(\mathbf{z}|\mathbf{x}) \\ &= \ell_q(\theta; \mathbf{x}) + \mathcal{H}(q) \end{aligned}$$

(this is where its name comes from)

- The first term is the expected complete log likelihood (energy) and the second term, which does not depend on  $\theta$ , is the entropy.
- Thus, in the M-step, maximizing with respect to  $\theta$  for fixed  $q$  we only need to consider the first term:

$$\theta^{t+1} = \operatorname{argmax}_\theta \ell_q(\theta; \mathbf{x}) = \operatorname{argmax}_\theta \sum_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log p(\mathbf{x}, \mathbf{z}|\theta)$$

- Claim: the optimum setting of  $q$  in the E-step is:

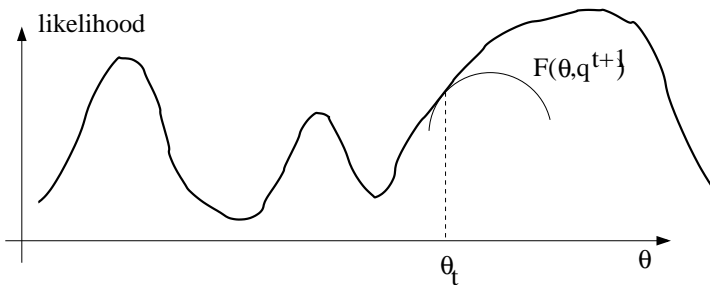
$$q^{t+1} = p(\mathbf{z}|\mathbf{x}, \theta^t)$$

- This is the posterior distribution over the latent variables given the data and the parameters. Often we need this at test time anyway (e.g. to perform classification).
- Proof (easy): this setting saturates the bound  $\ell(\theta; \mathbf{x}) \geq F(q, \theta)$

$$\begin{aligned} F(p(\mathbf{z}|\mathbf{x}, \theta^t), \theta^t) &= \sum_{\mathbf{z}} p(\mathbf{z}|\mathbf{x}, \theta^t) \log \frac{p(\mathbf{x}, \mathbf{z}|\theta^t)}{p(\mathbf{z}|\mathbf{x}, \theta^t)} \\ &= \sum_{\mathbf{z}} p(\mathbf{z}|\mathbf{x}, \theta^t) \log p(\mathbf{x}|\theta^t) \\ &= \log p(\mathbf{x}|\theta^t) \sum_{\mathbf{z}} p(\mathbf{z}|\mathbf{x}, \theta^t) \\ &= \ell(\theta; \mathbf{x}) \cdot 1 \end{aligned}$$

- Can also show this result using variational calculus or the fact that  $\ell(\theta) - F(q, \theta) = \text{KL}[q||p(\mathbf{z}|\mathbf{x}, \theta)]$

- Consider the likelihood function and the function  $F(q^{t+1}, \cdot)$ .



- Recall: a mixture of  $K$  Gaussians:

$$p(\mathbf{x}|\theta) = \sum_k \alpha_k \mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k)$$

$$\ell(\theta; \mathcal{D}) = \sum_n \log \sum_k \alpha_k \mathcal{N}(\mathbf{x}^n|\mu_k, \Sigma_k)$$

- Learning with EM algorithm:

$$\text{E-step: } p_{kn}^t = \mathcal{N}(\mathbf{x}^n|\mu_k^t, \Sigma_k^t)$$

$$q_{kn}^{t+1} = p(z=k|\mathbf{x}^n, \theta^t) = \frac{\alpha_k^t p_{kn}^t}{\sum_j \alpha_j^t p_{jn}^t}$$

$$\text{M-step: } \mu_k^{t+1} = \frac{\sum_n q_{kn}^{t+1} \mathbf{x}^n}{\sum_n q_{kn}^{t+1}}$$

$$\Sigma_k^{t+1} = \frac{\sum_n q_{kn}^{t+1} (\mathbf{x}^n - \mu_k^{t+1})(\mathbf{x}^n - \mu_k^{t+1})^\top}{\sum_n q_{kn}^{t+1}}$$

$$\alpha_k^{t+1} = \frac{1}{M} \sum_n q_{kn}^{t+1}$$

- A way of maximizing likelihood function for latent variable models. Finds ML parameters when the original (hard) problem can be broken up into two (easy) pieces:

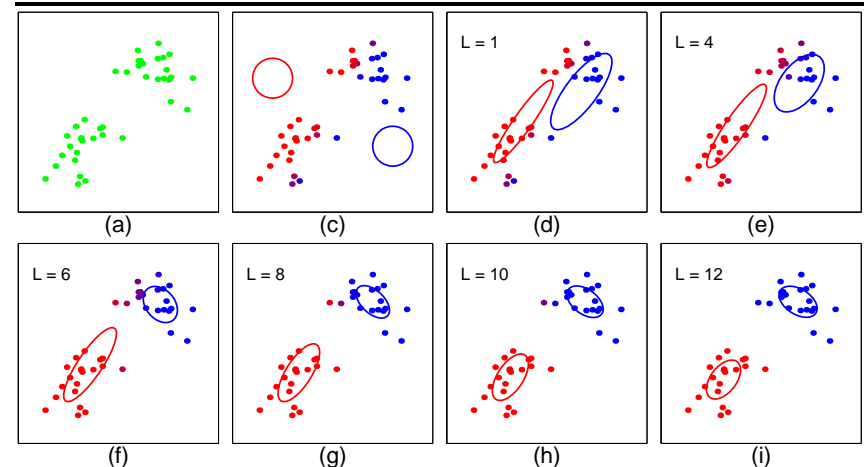
- Estimate some “missing” or “unobserved” data from observed data and current parameters.
- Using this “complete” data, find the maximum likelihood parameter estimates.

- Alternate between filling in the latent variables using our best guess (posterior) and updating the parameters based on this guess:

$$\text{E-step: } q^{t+1} = p(\mathbf{z}|\mathbf{x}, \theta^t)$$

$$\text{M-step: } \theta^{t+1} = \operatorname{argmax}_{\theta} \sum_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}) \log p(\mathbf{x}, \mathbf{z}|\theta)$$

- In the M-step we optimize a lower bound on the likelihood. In the E-step we close the gap, making bound=likelihood.



- Expected complete log likelihood  $\ell_q(\theta; \mathcal{D})$ :

$$\sum_n \sum_k q_{kn} \left[ \log \alpha_k - \frac{1}{2} (\mathbf{x}^n - \mu_k^{t+1})^\top \Sigma_k^{-1} (\mathbf{x}^n - \mu_k^{t+1}) - \frac{1}{2} \log |2\pi \Sigma_k| \right]$$

- For fixed  $q$  we can optimize the parameters:

$$\frac{\partial \ell_q}{\partial \mu_k} = \Sigma_k^{-1} \sum_n q_{kn} (\mathbf{x}^n - \mu_k)$$

$$\frac{\partial \ell_q}{\partial \Sigma_k^{-1}} = \frac{1}{2} \sum_n q_{kn} \left[ \Sigma_k^\top - (\mathbf{x}^n - \mu_k^{t+1})(\mathbf{x}^n - \mu_k^{t+1})^\top \right]$$

$$\frac{\partial \ell_q}{\partial \alpha_k} = \frac{1}{\alpha_k} \sum_n q_{kn} - \lambda \quad (\lambda = M)$$

- Fact:  $\frac{\partial \log |A^{-1}|}{\partial A^{-1}} = A^\top$  and  $\frac{\partial \mathbf{x}^\top A \mathbf{x}}{\partial A} = \mathbf{x} \mathbf{x}^\top$

- Of course, we can learn when there are missing (hidden) variables on some cases and not on others.

- In this case the cost function was:

$$\ell(\theta; \mathcal{D}) = \sum_{\text{complete}} \log p(\mathbf{x}^c, \mathbf{y}^c | \theta) + \sum_{\text{missing}} \log \sum_{\mathbf{y}} \log p(\mathbf{x}^m, \mathbf{y} | \theta)$$

- Now you can think of this in a new way: in the E-step we estimate the hidden variables on the incomplete cases only.
- The M-step optimizes the log likelihood on the complete data plus the expected likelihood on the incomplete data using the E-step.

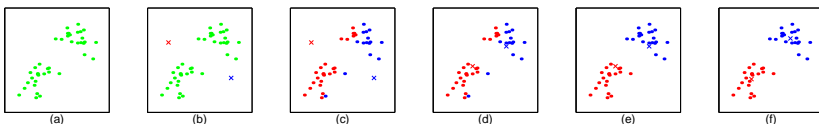
- The EM algorithm for mixtures of Gaussians is just like a soft version of the K-means algorithm.

- In the K-means “E-step” we do hard assignment:

$$c_n^{t+1} = \operatorname{argmin}_k (\mathbf{x}^n - \mu_k^t)^\top \Sigma_k^{-1} (\mathbf{x}^n - \mu_k^t)$$

- In the K-means “M-step” we update the means as the weighted sum of the data, but now the weights are 0 or 1:

$$\mu_k^{t+1} = \frac{\sum_n [c_k^{t+1} = n] \mathbf{x}^n}{\sum_n [c_k^{t+1} = n]}$$



- Some good things about EM:

- no learning rate parameter
- very fast for low dimensions
- each iteration guaranteed to improve likelihood
- adapts unused units rapidly

- Some bad things about EM:

- can get stuck in local minima
- both steps require considering *all* explanations of the data which is an exponential amount of work in the dimension of  $\theta$

- EM is typically used with mixture models, for example mixtures of Gaussians or mixtures of experts. The “missing” data are the labels showing which sub-model generated each datapoint.

Very common: also used to train HMMs, Boltzmann machines, ...

- 
- Sparse EM:  
Do not recompute exactly the posterior probability on each data point under all models, because it is almost zero. Instead keep an “active list” which you update every once in a while.
  - Generalized (Incomplete) EM: It might be hard to find the ML parameters in the M-step, even given the completed data. We can still make progress by doing an M-step that improves the likelihood a bit (e.g. gradient step).