Machine Learning

10-701/15-781, Fall 2006

Learning Graphical Models

Maximum Likelihood Estimation and Expectation Maximization

Eric Xing

Lecture 14, October 31, 2006

Reading: Chap. 1&2, C.B book

1 GM < BN < CI, Factorized: $P(x) = \prod P(x_i | \pi_i)$; MRF

2. $P(x_q | —)$, Z|;

2 Message Pass)

$P(X|\theta)$

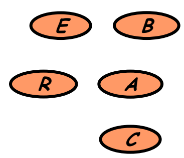
$\theta^* = \arg\max \log P(X|\theta)$

Learning Graphical Models

The goal:

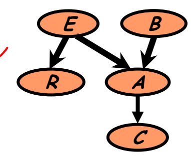
Given set of independent samples (***assignments*** of random variables), find the ***best*** (the most likely?) Bayesian Network (both DAG and CPDs)

n^2

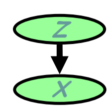


(B,E,A,C,R)=(T,F,F,T,F)
(B,E,A,C,R)=(T,F,T,T,F)
....
(B,E,A,C,R)=(F,T,T,T,F)

2^{n^2}



E	B	P(A | E,B)	
e	b	0.9	0.1
e	$\bar{b}$	0.2	0.8
$\bar{e}$	b	0.9	0.1
$\bar{e}$	$\bar{b}$	0.01	0.99



Learning completely observed GMs

- The data:

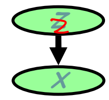
  $\{(z^{(1)},x^{(1)}), (z^{(2)},x^{(2)}), (z^{(3)},x^{(3)}), \ldots (z^{(N)},x^{(N)})\}$

Review: the basic idea underlying MLE

- The completely observed model:

$p(X, Z)$

$= p(X|Z)p(Z)$



  - Z is a class indicator vector

$$Z = \begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_M \end{bmatrix}, \qquad \text{where } Z_m = [0,1], \text{ and } \sum_m Z_m = 1$$

and a datum is in class i w.p. π_i

$$p(z_i = 1 \mid \pi) = \pi_i = \pi_1^{z_1} \times \pi_2^{z_2} \times \ldots \times \pi_M^{z_M}$$

All except one of these terms will be one

$$p(z) = \prod_m \pi_m^{z_m}$$

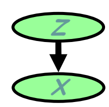
  - X is a conditional Gaussian variable with a class-specific mean

$$p(x \mid z_m = 1, \mu, \sigma) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\tfrac{1}{2\sigma^2}(x - \mu_m)^2\right\}$$

$$p(x \mid z, \mu, \sigma) = \prod_m N(x \mid \mu_m, \sigma)^{z_m}$$

Review: the basic idea underlying MLE

- Data log-likelihood $\prod P(Z,X)$



$$\begin{aligned} l(\boldsymbol{\theta} \mid D) &= \log \prod_n p(z^{(n)}, x^{(n)}) = \log \prod_n p(z^{(n)} \mid \pi) p(x^{(n)} \mid z^{(n)}, \mu, \sigma) \\ &= \sum_n \log p(z^{(n)} \mid \pi) + \sum_n \log p(x^{(n)} \mid z^{(n)}, \mu, \sigma) \\ &= \sum_n \log \prod_m \pi_m^{z_m^{(n)}} + \sum_n \log \prod_m N(x^{(n)} \mid \mu_m, \sigma)^{z_m^{(n)}} \\ &= \sum_n \sum_m z_m^{(n)} \log \pi_m - \sum_n \sum_m z_m^{(n)} \tfrac{1}{2\sigma^2}(x^{(n)} - \mu_m)^2 + C \end{aligned}$$

- MLE

$$\pi_m^* = \arg\max l(\boldsymbol{\theta} \mid D), \qquad \Rightarrow \frac{\partial}{\partial \pi_m} l(\boldsymbol{\theta} \mid D) = 0, \forall m, \quad \text{s.t.} \sum_m \pi_m = 1$$

$$\Rightarrow \pi_m^* = \sum_n z_m^{(n)} \Big/ N = n_m \Big/ N$$

the fraction of samples of class m

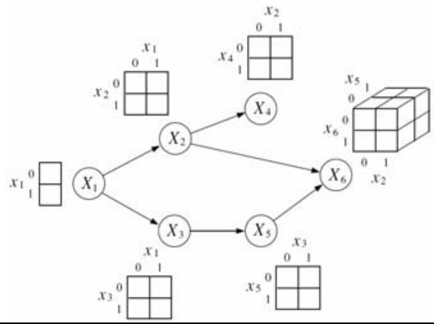
$$\mu_m^* = \arg\max l(\boldsymbol{\theta} \mid D), \qquad \Rightarrow \mu_m^* = \frac{\sum_n z_m^{(n)} x^{(n)}}{\sum_n z_m^{(n)}} = \frac{\sum_n z_m^{(n)} x^{(n)}}{n_m}$$

the average of samples of class m

MLE for general BNs

- If we assume the parameters for each CPD are globally independent, and all nodes are fully observed, then the log-likelihood function decomposes into a sum of local terms, one per node:

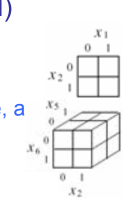
$$\ell(\theta;D) = \log p(D\mid\theta) = \log\prod_n\left(\prod_i p(x_{n,i}\mid \mathbf{x}_{n,\pi_i},\theta_i)\right) = \sum_i\left(\sum_n \log p(x_{n,i}\mid \mathbf{x}_{n,\pi_i},\theta_i)\right)$$



MLE for BNs with tabular CPDs

- Assume each CPD is represented as a table (multinomial) where

$$\theta_{ijk} \overset{\text{def}}{=} p(X_i = j \mid X_{\pi_i} = k)$$



  - Note that in case of multiple parents, $\mathbf{X}_{\pi_i}$ will have a composite state, a CPD will be a high-dimensional table
  - The sufficient statistics are counts of family configurations

$$n_{ijk} \overset{\text{def}}{=} \sum_n x_{n,i}^j x_{n,\pi_i}^k$$

- The log-likelihood is

$$\ell(\theta;D) = \log\prod_{i,j,k}\theta_{ijk}^{n_{ijk}} = \sum_{i,j,k} n_{ijk}\log\theta_{ijk}$$

- Using a Lagrange multiplier to enforce so $\sum_j \theta_{ijk} = 1$ we get

$$\theta_{ijk}^{ML} = \frac{n_{ijk}}{\sum_{j'} n_{ij'k}}$$

Partially observed GMs

- Speech recognition

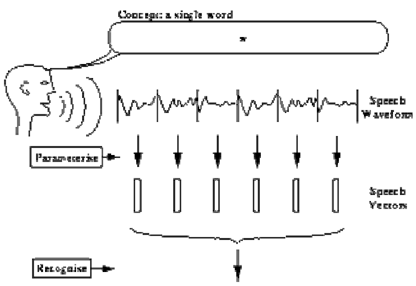
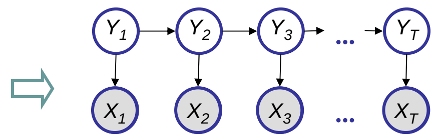
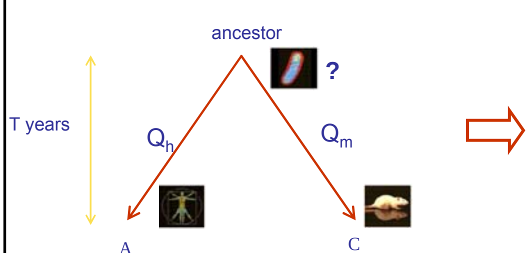


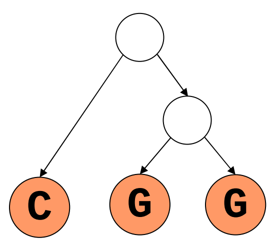
Fig. 1.2 Isolated Word Problem



Partially observed GM

- Biological Evolution

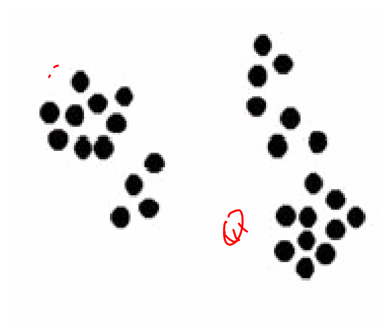




Unobserved Variables

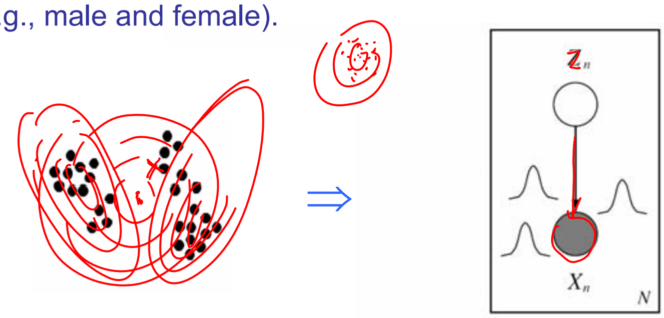
- A variable can be unobserved (latent) because:
  - it is an imaginary quantity meant to provide some simplified and abstractive view of the data generation process
    - e.g., speech recognition models, mixture models …
  - it is a real-world object and/or phenomena, but difficult or impossible to measure
    - e.g., the temperature of a star, causes of a disease, evolutionary ancestors …
  - it is a real-world object and/or phenomena, but sometimes wasn't measured, because of faulty sensors; or was measure with a noisy channel, etc.
    - e.g., traffic radio, aircraft signal on a radar screen,
- Discrete latent variables can be used to partition/cluster data into sub-groups (mixture models, forthcoming).
- Continuous latent variables (factors) can be used for dimensionality reduction (factor analysis, etc., later lectures).

Mixture Models



Mixture Models, con'd

- A density model $p(x)$ may be multi-modal.
- We may be able to model it as a mixture of uni-modal distributions (e.g., Gaussians).
- Each mode may correspond to a different sub-population (e.g., male and female).



Gaussian Mixture Models (GMMs)

- Consider a mixture of K Gaussian components:
  - Z is a latent class indicator vector:

$$p(z_n) = \mathrm{multi}(z_n : \pi) = \prod_k (\pi_k)^{z_n^k}$$

  - X is a conditional Gaussian variable with a class-specific mean/covariance

$$p(x_n \mid z_n^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{m/2}|\Sigma_k|^{1/2}} \exp\left\{-\tfrac{1}{2}(x_n - \mu_k)^T \Sigma_k^{-1}(x_n - \mu_k)\right\}$$

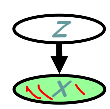
$p(z_{nk} | x) = \pi_k N(x | \mu_k, \Sigma_k)$

  - The likelihood of a sample:

mixture component

mixture proportion

$$p(x_n | \mu, \Sigma) = \sum_k p(z^k = 1 | \pi) p(x, | z^k = 1, \mu, \Sigma)$$
$$= \sum_{z_n} \prod_k \left((\pi_k)^{z_n^k} N(x_n : \mu_k, \Sigma_k)^{z_n^k}\right) = \sum_k \pi_k N(x, | \mu_k, \Sigma_k)$$

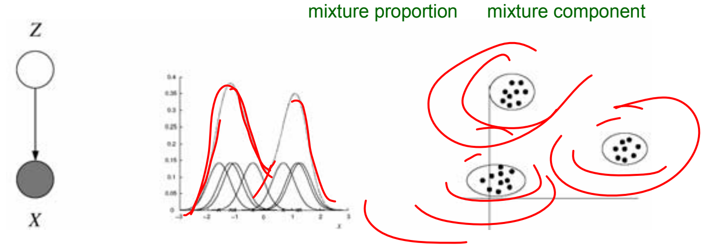


Gaussian Mixture Models (GMMs)

- Consider a mixture of K Gaussian components:

$$p(x_n|\mu,\Sigma)=\sum_k \pi_k N(x,|\mu_k,\Sigma_k)$$

mixture proportion   mixture component



- This model can be used for unsupervised clustering.
  - This model (fit by AutoClass) has been used to discover new kinds of stars in astronomical data, etc.

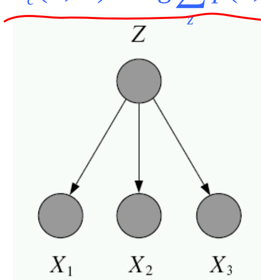
Why is Learning Harder?

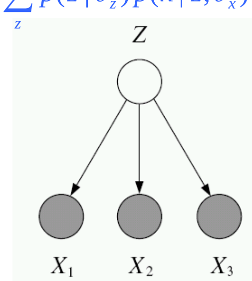
- In fully observed iid settings, the log likelihood decomposes into a sum of local terms (at least for directed models).

$$\ell_c(\theta;D)=\log p(x,z|\theta)=\log p(z|\theta_z)+\log p(x|z,\theta_x)$$

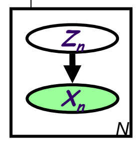
- With latent variables, all the parameters become coupled together via marginalization

$$\ell_c(\theta;D)=\log\sum_z p(x,z|\theta)=\log\sum_z p(z|\theta_z)p(x|z,\theta_x)$$





Toward the EM algorithm



- E.g., A mixture of K Gaussians:
  - Z is a latent class indicator vector

$$p(z_n) = \text{multi}(z_n : \pi) = \sum_k (\pi_k)^{z_n^k}$$

  - X is a conditional Gaussian variable with a class-specific mean/covariance

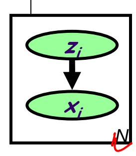
$$p(x_n \mid z_n^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{m/2} |\Sigma_k|^{1/2}} \exp\left\{ -\tfrac{1}{2} (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right\}$$

  - The likelihood of a sample:

$$\begin{aligned} p(x_n | \mu, \Sigma) &= \sum_k p(z^k = 1 \mid \pi) p(x, \mid z^k = 1, \mu, \Sigma) \\ &= \sum_{z_n} \prod_k \left((\pi_k)^{z_n^k} N(x_n : \mu_k, \Sigma_k)^{z_n^k} \right) = \sum_k \pi_k N(x, \mid \mu_k, \Sigma_k) \end{aligned}$$

Eric Xing 17

Toward the EM algorithm



- Recall MLE for completely observed data
- Data log-likelihood

$$\begin{aligned} \ell(\boldsymbol{\theta}; D) &= \log \sum_n p(z_n, x_n) = \log \prod_n p(z_n \mid \pi) p(x_n \mid z_n, \mu, \sigma) \\ &= \sum_n \log \prod_k \pi_k^{z_n^k} + \sum_n \log \prod_k N(x_n; \mu_k, \sigma)^{z_n^k} \\ &= \sum_n \sum_k z_n^k \log \pi_k - \sum_n \sum_k z_n^k \tfrac{1}{2\sigma^2} (x_n - \mu_k)^2 + C \end{aligned}$$

- MLE

$$\begin{aligned} \hat{\pi}_{k,MLE} &= \arg\max_\pi \ell(\boldsymbol{\theta}; D), \\ \hat{\mu}_{k,MLE} &= \arg\max_\mu \ell(\boldsymbol{\theta}; D) \qquad \Rightarrow \quad \hat{\mu}_{k,MLE} = \frac{\sum_n z_n^k x_n}{\sum_n z_n^k} \\ \hat{\sigma}_{k,MLE} &= \arg\max_\sigma \ell(\boldsymbol{\theta}; D) \end{aligned}$$

- What if we do not know z_n?

Eric Xing 18

Expectation-Maximization (EM) Algorithm

- EM is an optimization strategy for objective functions that can be interpreted as likelihoods in the presence of missing data.
- It is much simpler than gradient methods:
  - No need to choose step size.
  - Enforces constraints automatically.
  - Calls inference and fully observed learning as subroutines.
- EM is an Iterative algorithm with two linked steps:
  - E-step: fill-in hidden values using inference, $p(z|x, \theta)$.
  - M-step: update parameters t+1 using standard MLE/MAP method applied to completed data
- We will (hopefully) prove that this procedure monotonically improves (or leaves it unchanged). Thus it always converges to a local optimum of the likelihood.

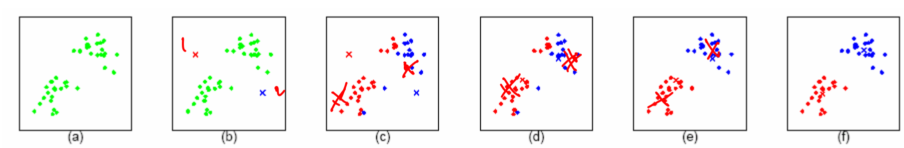
K-means

- Start:
  - "Guess" the centroid μ_k and coveriance Σ_k of each of the K clusters
- Loop
  - For each point n=1 to N,
    compute its cluster label:

$$z_n^{(t)} = \arg\max_k (x_n - \mu_k^{(t)})^T \Sigma_k^{-1(t)} (x_n - \mu_k^{(t)})$$

  - For each cluster k=1:K

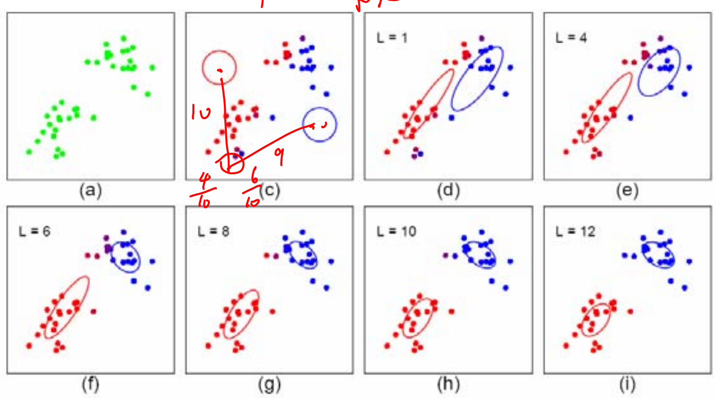
$$\mu_k^{(t+1)} = \frac{\sum_n \delta(z_n^{(t)}, k) x_n}{\sum_n \delta(z_n^{(t)}, k)} \qquad \Sigma_k^{(t+1)} = \ldots$$



Expectation-Maximization

- Start:
  - "Guess" the centroid μ_k and coveriance Σ_k of each of the K clusters
- Loop

$\mu = \frac{x_n \delta(z=k)}{N_k}$ $\{\mu_k, \Sigma_k\}$ $\mu =$



Example: Gaussian mixture model

- A mixture of K Gaussians:
  - Z is a latent class indicator vector

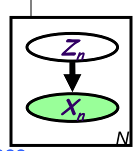
$$p(z_n) = \text{multi}(z_n : \pi) = \sum_k (\pi_k)^{z_n^k}$$

  - X is a conditional Gaussian variable with a class-specific mean/covariance

$$p(x_n \mid z_n^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{m/2}|\Sigma_k|^{1/2}} \exp\left\{-\tfrac{1}{2}(x_n - \mu_k)^T \Sigma_k^{-1}(x_n - \mu_k)\right\}$$

  - The likelihood of a sample:

$$p(x_n|\mu,\Sigma) = \sum_k p(z^k = 1 \mid \pi) p(x, | z^k = 1, \mu, \Sigma)$$
$$= \sum_{z_n} \prod_k \left((\pi_k)^{z_n^k} N(x_n : \mu_k, \Sigma_k)^{z_n^k}\right) = \sum_k \pi_k N(x \mid \mu_k, \Sigma_k)$$



- The expected complete log likelihood $\langle \log P(x, z \mid \theta) \rangle_{q(z)}$

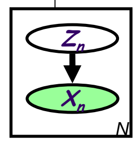
$$\langle \ell_c(\theta; x, z) \rangle = \sum_n \langle \log p(z_n \mid \pi) \rangle_{p(z|x)} + \sum_n \langle \log p(x_n \mid z_n, \mu, \Sigma) \rangle_{p(z|x)}$$

$$= \sum_n \sum_k \langle z_n^k \rangle \log \pi_k - \frac{1}{2} \sum_n \sum_k \langle z_n^k \rangle \left((x_n - \mu_k)^T \Sigma_k^{-1}(x_n - \mu_k) + \log|\Sigma_k| + C\right)$$

$\langle \log P(z) + \log P(x|z) \rangle_{q(z)}$

E-step

- We maximize $\langle l_c(\theta) \rangle$ iteratively using the following iterative procedure:



  – Expectation step: computing the expected value of the ~~sufficient statistics~~ of the hidden variables (i.e., z) given current est. of the parameters (i.e., π and μ).

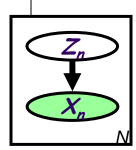
$$\tau_n^{k(t)} = \langle z_n^k \rangle_{q^{(t)}} = p(z_n^k = 1 | x, \mu^{(t)}, \Sigma^{(t)}) = \frac{\pi_k^{(t)} N(x_n, | \mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_i \pi_i^{(t)} N(x_n, | \mu_i^{(t)}, \Sigma_i^{(t)})},$$

$$\frac{p(z,x)}{\sum_z p(z,x)}$$

  - Here we are essentially doing **inference**

Recall out objective

- The expected complete log likelihood

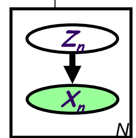


$$\langle \ell_c(\theta; x, z) \rangle = \sum_n \langle \log p(z_n | \pi) \rangle_{p(z|x)} + \sum_n \langle \log p(x_n | z_n, \mu, \Sigma) \rangle_{p(z|x)}$$

$$= \sum_n \sum_k \langle z_n^k \rangle \log \pi_k - \frac{1}{2} \sum_n \sum_k \langle z_n^k \rangle \left((x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) + \log|\Sigma_k| + C \right)$$

M-step

- We maximize $\langle l_c(\boldsymbol{\theta})\rangle$ iteratively using the following iterative procudure:
  - Maximization step: compute the parameters under current results of the expected value of the hidden variables



$$\pi_k^* = \arg\max\langle l_c(\boldsymbol{\theta})\rangle, \quad \Rightarrow \frac{\partial}{\partial \pi_k}\langle l_c(\boldsymbol{\theta})\rangle = 0, \forall k, \quad \text{s.t.} \sum_k \pi_k = 1 \quad \Sigma \qquad \text{MLE}$$

$$\Rightarrow \pi_k^* = \sum_n \langle z_n^k \rangle_{q^{(t)}} \Big/ N = \sum_n \tau_n^{k(t)} \Big/ N = \langle n_k \rangle \Big/ N$$

z_n^k

$$\mu_k^* = \arg\max\langle l(\boldsymbol{\theta})\rangle, \quad \Rightarrow \mu_k^{(t+1)} = \frac{\sum_n \tau_n^{k(t)} x_n}{\sum_n \tau_n^{k(t)}}$$

$$\Sigma_k^* = \arg\max\langle l(\boldsymbol{\theta})\rangle, \quad \Rightarrow \Sigma_k^{(t+1)} = \frac{\sum_n \tau_n^{k(t)} (x_n - \mu_k^{(t+1)})(x_n - \mu_k^{(t+1)})^T}{\sum_n \tau_n^{k(t)}}$$

Fact:

$$\frac{\partial \log|\mathbf{A}^{-1}|}{\partial \mathbf{A}^{-1}} = \mathbf{A}^T$$

$$\frac{\partial \mathbf{x}^T \mathbf{A}\mathbf{x}}{\partial \mathbf{A}} = \mathbf{x}\mathbf{x}^T$$

- This is isomorphic to **MLE** except that the variables that are hidden are replaced by their expectations (in general they will by replaced by their corresponding "**sufficient statistics**")

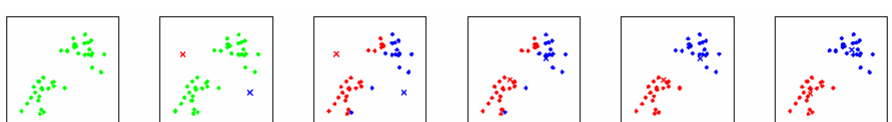
Compare: K-means

- The EM algorithm for mixtures of Gaussians is like a "soft version" of the K-means algorithm.
- In the K-means "E-step" we do hard assignment:

$$z_n^{(t)} = \arg\max_k (x_n - \mu_k^{(t)})^T \Sigma_k^{-1(t)} (x_n - \mu_k^{(t)})$$

- 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:

$$\mu_k^{(t+1)} = \frac{\sum_n \delta(z_n^{(t)}, k) x_n}{\sum_n \delta(z_n^{(t)}, k)} \qquad p(z|x)$$



(a) (b) (c) (d) (e) (f)

EM for general BNs

while not converged

% E-step

for each node i

$ESS_i = 0$ % reset expected sufficient statistics

for each data sample n

do inference with $X_{n,H}$

for each node i

$$ESS_i \mathrel{+}= \left\langle SS_i(x_{n,i}, x_{n,\pi_i}) \right\rangle_{p(x_{n,H}|x_{n,-H})}$$

% M-step

for each node i

$\theta_i :=$ MLE(ESS_i)

Partially Hidden Data

- Of course, we can learn when there are missing (hidden) variables on some cases and not on others.
- In this case the cost function is:

$$\ell_c(\theta; D) = \sum_{n \in \text{Complete}} \log p(x_n, y_n \mid \theta) + \sum_{m \in \text{Missing}} \log \sum_{y_m} p(x_m, y_m \mid \theta)$$

  - Note that y_m do not have to be the same in each case --- the data can have different missing values in each different sample

- 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.

Optional Material!

-- Theory underlying EM

Theory underlying EM

- What are we doing?
- Recall that according to MLE, we intend to learn the model parameter that would have maximize the likelihood of the data.
- But we do not observe z, so computing

$$\ell_c(\theta; D) = \log \sum_z p(x, z \mid \theta) = \log \sum_z p(z \mid \theta_z) p(x \mid z, \theta_x)$$

is difficult!

- What shall we do?

Complete & Incomplete Log Likelihoods

- Complete log likelihood

  Let X denote the observable variable(s), and Z denote the latent variable(s). If Z could be observed, then

  $$\ell_c(\theta; x, z) \overset{\text{def}}{=} \log p(x, z \mid \theta)$$

  - Usually, optimizing $\ell_c()$ given both z and x is straightforward (c.f. MLE for fully observed models).
  - Recalled that in this case the objective for, e.g., MLE, decomposes into a sum of factors, the parameter for each factor can be estimated separately.
  - **But given that Z is not observed, $\ell_c()$ is a random quantity, cannot be maximized directly**.

- Incomplete log likelihood

  With z unobserved, our objective becomes the log of a marginal probability:

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

  - **This objective won't decouple**

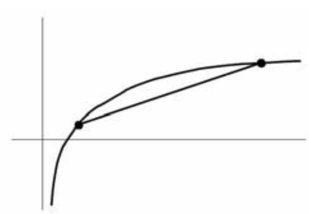
Expected Complete Log Likelihood

- For ***any*** distribution $q(z)$, define *expected complete log likelihood*:

  $$\langle \ell_c(\theta; x, z) \rangle_q \overset{\text{def}}{=} \sum_z q(z \mid x, \theta) \log p(x, z \mid \theta)$$

  - A deterministic function of θ
  - Linear in $\ell_c()$ --- inherit its factorizabiility
  - Does maximizing this surrogate yield a maximizer of the likelihood?

- Jensen's inequality



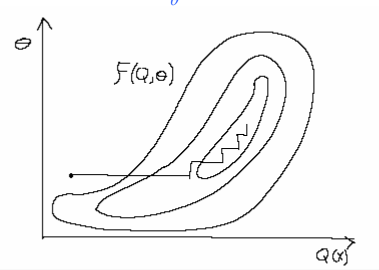
$$\begin{aligned}
\ell(\theta; x) &= \log p(x \mid \theta) \\
&= \log \sum_z p(x, z \mid \theta) \\
&= \log \sum_z q(z \mid x) \frac{p(x, z \mid \theta)}{q(z \mid x)} \\
&\geq \sum_z q(z \mid x) \log \frac{p(x, z \mid \theta)}{q(z \mid x)}
\end{aligned} \quad \Rightarrow \quad \ell(\theta; x) \geq \langle \ell_c(\theta; x, z) \rangle_q + H_q$$

Lower Bounds and Free Energy

- For fixed data x, define a functional called the free energy:

$$F(q,\theta) \overset{\text{def}}{=} \sum_z q(z\,|\,x)\log\frac{p(x,z\,|\,\theta)}{q(z\,|\,x)} \le \ell(\theta;x)$$

- The EM algorithm is coordinate-ascent on F:
  - **E-step:** $q^{t+1} = \arg\max_q F(q,\theta^t)$
  - **M-step:** $\theta^{t+1} = \arg\max_\theta F(q^{t+1},\theta^t)$



E-step: maximization of expected ℓ_c w.r.t. q

- Claim: $q^{t+1} = \arg\max_q F(q,\theta^t) = p(z\,|\,x,\theta^t)$
  - 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 attains the bound $\ell(\theta;x)\ge F(q,\theta)$

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

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

E-step ≡ plug in posterior expectation of latent variables

- Without loss of generality: assume that $p(x,z|\theta)$ is a generalized exponential family distribution:

$$p(x,z|\theta) = \frac{1}{Z(\theta)} h(x,z) \exp\left\{\sum_i \theta_i f_i(x,z)\right\}$$

- The expected complete log likelihood under $q^{t+1} = p(z|x,\theta^t)$ is

$$\begin{aligned}\langle \ell_c(\theta^t;x,z)\rangle_{q^{t+1}} &= \sum_z q(z|x,\theta^t)\log p(x,z|\theta^t) - A(\theta) \\ &= \sum_i \theta_i^t \langle f_i(x,z)\rangle_{q(z|x,\theta^t)} - A(\theta)\end{aligned}$$

M-step: maximization of expected ℓ_c w.r.t. θ

- Note that the free energy breaks into two terms:

$$\begin{aligned}F(q,\theta) &= \sum_z q(z|x)\log\frac{p(x,z|\theta)}{q(z|x)} \\ &= \sum_z q(z|x)\log p(x,z|\theta) - \sum_z q(z|x)\log q(z|x) \\ &= \langle \ell_c(\theta;x,z)\rangle_q + H_q\end{aligned}$$

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

$$\theta^{t+1} = \arg\max_\theta \langle \ell_c(\theta;x,z)\rangle_{q^{t+1}} = \arg\max_\theta \sum_z q(z|x)\log p(x,z|\theta)$$

  - Under optimal q^{t+1}, this is equivalent to solving a standard MLE of fully observed model $p(x,z|\theta)$, with the sufficient statistics involving z replaced by their expectations w.r.t. $p(z|x,\theta)$.

Summary: EM Algorithm

- A way of maximizing likelihood function for latent variable models. Finds MLE of parameters when the original (hard) problem can be broken up into two (easy) pieces:
  1. Estimate some "missing" or "unobserved" data from observed data and current parameters.
  2. Using this "complete" data, find the maximum likelihood parameter estimates.

- Alternate between filling in the latent variables using the best guess (posterior) and updating the parameters based on this guess:
  - E-step: $q^{t+1} = \arg\max_q F(q, \theta^t)$
  - M-step: $\theta^{t+1} = \arg\max_\theta F(q^{t+1}, \theta^t)$

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

A Report Card for EM

- Some good things about EM:
  - no learning rate (step-size) parameter
  - automatically enforces parameter constraints
  - very fast for low dimensions
  - each iteration guaranteed to improve likelihood

- Some bad things about EM:
  - can get stuck in local minima
  - can be slower than conjugate gradient (especially near convergence)
  - requires expensive inference step
  - is a maximum likelihood/MAP method