Probabilistic Graphical Models

10-708

Learning Partially Observed Graphical Models

Eric Xing

Lecture 13, Oct 26, 2005

Reading: MJ-Chap. 5,10,11

Partially observed GMs

- Speech recognition

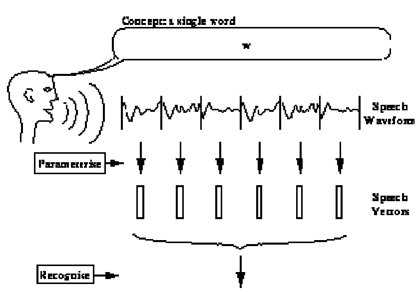
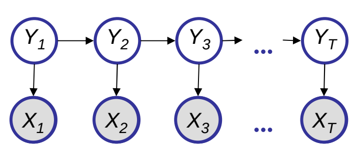
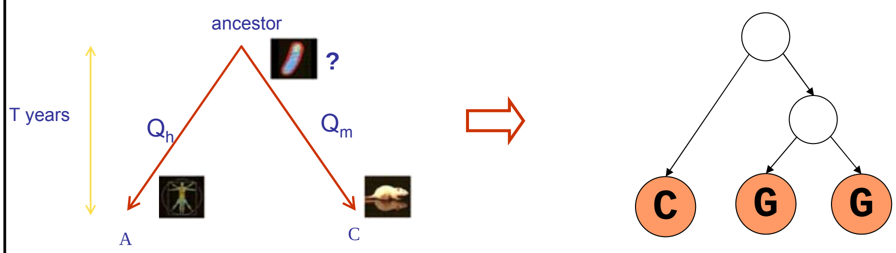


Fig. 1.2 Isolated Word Problem



Partially observed GM

- Biological Evolution

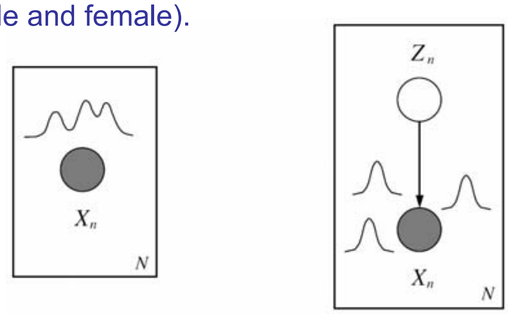


Unobserved Variables

- 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, etc.
- Discrete latent variables can be used to partition/cluster data into sub-groups.
- Continuous latent variables (factors) can be used for dimensionality reduction (factor analysis, etc).

Mixture models

- 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).



(a) (b)

Gaussian Mixture Models (GMMs)

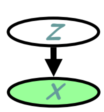
- Consider a mixture of K Gaussian components:
  - Z is a latent class indicator vector: $p(z_n) = \text{multi}(z_n : \pi) = \prod_k (\pi_k)^{z_n^k}$
  - X is a conditional Gaussian variable with a class-specific mean/covariance

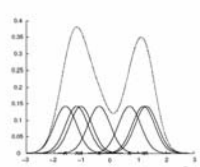
$$p(x_n \mid z_n^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{m/2} |\Sigma_k|^{1/2}} \exp\left\{ -\tfrac{1}{2} (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right\}$$

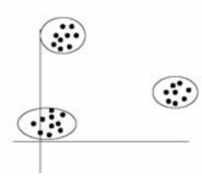
  - The likelihood of a sample:

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

mixture proportion — mixture component

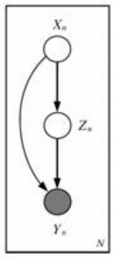


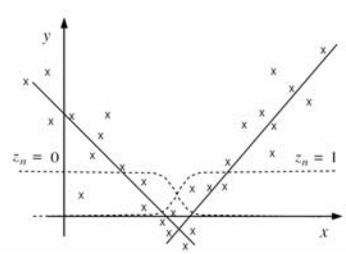




- This model can be used for unsupervised clustering.
  - This model (fit by AutoClass) has been used to discover new kinds of stars in astronomical data, etc.

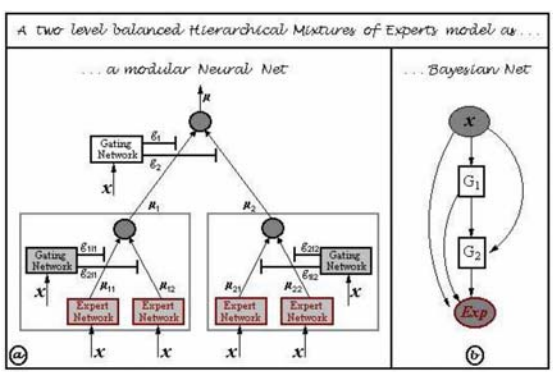
Conditional mixture model: Mixture of experts





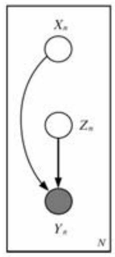
- We will model $p(Y|X)$ using different experts, each responsible for different regions of the input space.
  - Latent variable Z chooses expert using softmax gating function:
    $$P(z^k = 1|x) = \mathrm{Softmax}(\xi^T x)$$
  - Each expert can be a linear regression model: $P(y|x, z^k = 1) = \mathcal{N}(y; \theta_k^T x, \sigma_k^2)$
  - The posterior expert responsibilities are
    $$P(z^k = 1|x, y, \theta) = \frac{p(z^k = 1|x) p_k(y|x, \theta_k, \sigma_k^2)}{\sum_j p(z^j = 1|x) p_j(y|x, \theta_j, \sigma_j^2)}$$

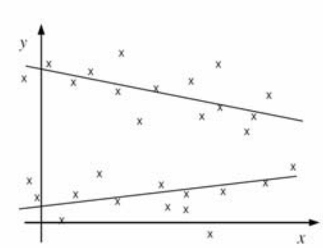
Hierarchical mixture of experts



- This is like a soft version of a depth-2 classification/regression tree.
- $P(Y|X,G_1,G_2)$ can be modeled as a GLIM, with parameters dependent on the values of G_1 and G_2 (which specify a "conditional path" to a given leaf in the tree).

Mixture of overlapping experts





- By removing the $X \rightarrow Z$ arc, we can make the partitions independent of the input, thus allowing overlap.
- This is a mixture of linear regressors; each subpopulation has a different conditional mean.

$$P(z^k = 1 | x, y, \theta) = \frac{p(z^k = 1) p_k(y | x, \theta_k, \sigma_k^2)}{\sum_j p(z^j = 1) p_j(y | x, \theta_j, \sigma_j^2)}$$

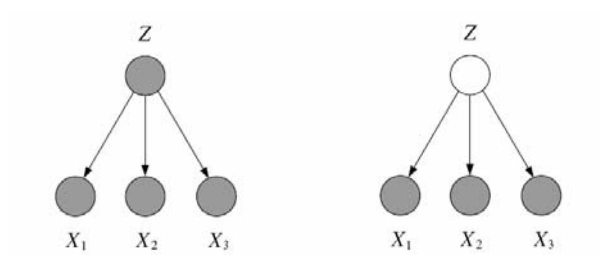
Why is Learning Harder?

- 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

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



Gradient Learning for mixture models

- We can learn mixture densities using gradient descent on the log likelihood. The gradients are quite interesting:

$$\ell(\theta) = \log p(\mathrm{x} \mid \theta) = \log \sum_k \pi_k p_k(\mathrm{x}|\theta_k)$$

$$\frac{\partial \ell}{\partial \theta} = \frac{1}{p(\mathrm{x} \mid \theta)} \sum_k \pi_k \frac{\partial p_k(\mathrm{x}|\theta_k)}{\partial \theta}$$

$$= \sum_k \frac{\pi_k}{p(\mathrm{x} \mid \theta)} p_k(\mathrm{x}|\theta_k) \frac{\partial \log p_k(\mathrm{x}|\theta_k)}{\partial \theta}$$

$$= \sum_k \pi_k \frac{p_k(\mathrm{x}|\theta_k)}{p(\mathrm{x} \mid \theta)} \frac{\partial \log p_k(\mathrm{x}|\theta_k)}{\partial \theta_k} = \sum_k r_k \frac{\partial \ell_k}{\partial \theta_k}$$

- In other words, the gradient is the responsibility weighted sum of the individual log likelihood gradients.
- Can pass this to a conjugate gradient routine.

Parameter Constraints

- Often we have constraints on the parameters, e.g. $\Sigma_k \pi_k = 1$, Σ being symmetric positive definite (hence $\Sigma_{ii} > 0$).
- We can use constrained optimization, or we can reparameterize in terms of unconstrained values.
  - For normalized weights, use the softmax transform: $\pi_k = \frac{\exp(\gamma_k)}{\Sigma_j \exp(\gamma_j)}$
  - For covariance matrices, use the Cholesky decomposition:

$$\Sigma^{-1} = \mathbf{A}^T \mathbf{A}$$

where A is upper diagonal with positive diagonal:

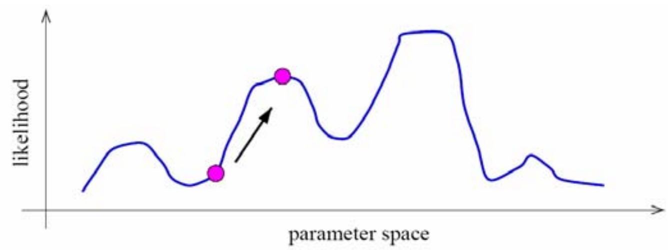
$$\mathbf{A}_{ii} = \exp(\lambda_i) > 0 \quad \mathbf{A}_{ij} = \eta_{ij} \ \ (j > i) \quad \mathbf{A}_{ij} = 0 \ \ (j < i)$$

the parameters $\gamma_i, \lambda_i, \eta_{ij} \in \mathbb{R}$ are unconstrained.

  - Use chain rule to compute $\frac{\partial \ell}{\partial \pi}, \frac{\partial \ell}{\partial \mathbf{A}}$.

Identifiability

- A mixture model induces a multi-modal likelihood.
- Hence gradient ascent can only find a local maximum.
- Mixture models are unidentifiable, since we can always switch the hidden labels without affecting the likelihood.
- Hence we should be careful in trying to interpret the "meaning" of latent variables.



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 prove that this procedure monotonically improves (or leaves it unchanged). Thus it always converges to a local optimum of the likelihood.

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

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

  - 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**

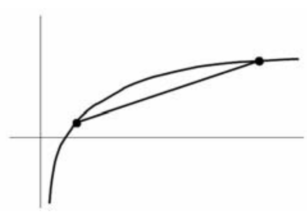
Expected Complete Log Likelihood

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

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

  - 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



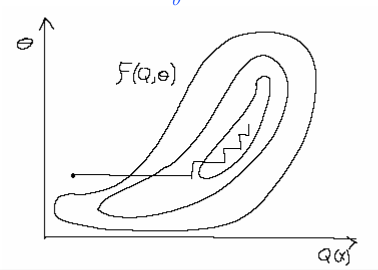
$$\begin{aligned}\ell(\theta; x) &= \log p(x \mid \theta) \\ &= \log \sum_z p(x, z \mid \theta) \\ &= \log \sum_z q(z \mid x) \frac{p(x, z \mid \theta)}{q(z \mid x)} \\ &= \sum_z q(z \mid x) \log \frac{p(x, z \mid \theta)}{q(z \mid x)} \qquad \Rightarrow \quad \ell(\theta; x) \geq \langle \ell_c(\theta;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

Lower Bounds and Free Energy

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

$$F(q,\theta) \stackrel{\text{def}}{=} \sum_z q(z \mid x) \log \frac{p(x,z \mid \theta)}{q(z \mid x)} \leq \ell(\theta; x)$$

- 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: $q^{t+1} = \arg\max_q F(q, \theta^t) = p(z \mid x, \theta^t)$
  - 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 $\ell(\theta;x) \geq F(q,\theta)$

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

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

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:

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

  - Special cases: if $p(X|Z)$ are GLIMs, then $f_i(x,z) = \eta_i^T(z)\xi_i(x)$

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

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

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

- Note that the free energy breaks into two terms:

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

  - The first term is the expected complete log likelihood (energy) and the second term, which does not depend on θ, is the entropy.

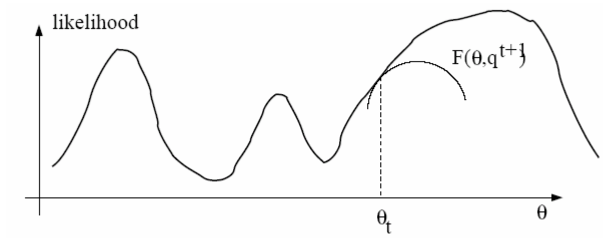
- Thus, in the M-step, maximizing with respect to θ for fixed q we only need to consider the first term:

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

  - 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. $p(z|x,\theta)$.

EM Constructs Sequential Convex Lower Bounds

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



- A hill-climbing algorithm

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.

Example: Gaussian mixture model

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

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

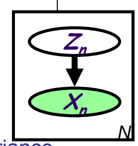
  - X is a conditional Gaussian variable with a class-specific mean/covariance

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

  - The likelihood of a sample:

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

GM:

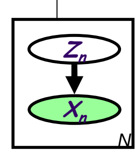


- The expected complete log likelihood

$$\langle \ell_c(\theta; x, z)\rangle = \sum_n \langle \log p(z_n \mid \pi)\rangle_{p(z|x)} + \sum_n \langle \log p(x_n \mid z_n, \mu, \Sigma)\rangle_{p(z|x)}$$
$$= \sum_n \sum_k \langle z_n^k \rangle \log \pi_k - \frac{1}{2} \sum_n \sum_k \langle z_n^k \rangle \left((x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) + \log|\Sigma_k| + C\right)$$

E-step

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



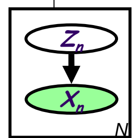
  – Expectation step: computing the expected value of the hidden variables (i.e., z) given current est. of the parameters (i.e., π and μ).

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

  - Here we are essentially doing **inference**

M-step

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



  – Maximization step: compute the parameters under current results of the expected value of the hidden variables

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

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

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

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

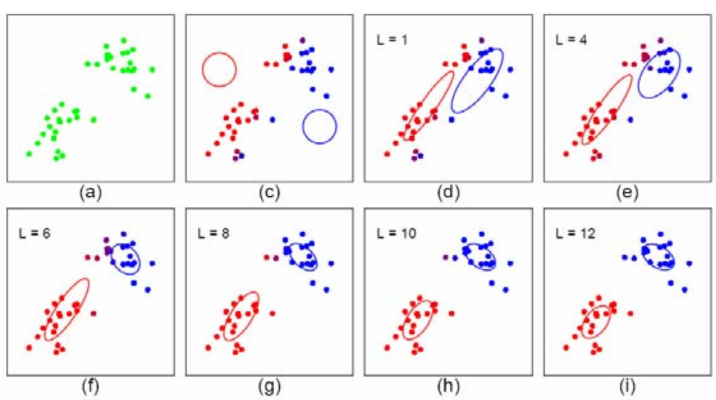
Fact:

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

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

  - 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**")

EM for MOG



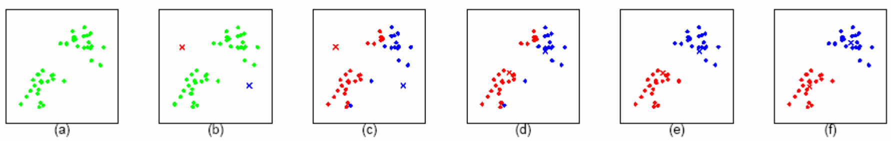
Compare: K-means

- 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:

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

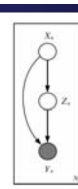


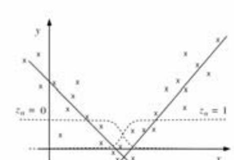
(a) (b) (c) (d) (e) (f)

EM for conditional mixture model

- Model:

$$P(y|x) = \sum_k p(z^k = 1 \mid x, \xi) p(y \mid z^k = 1, x, \theta_i, \sigma)$$





- The objective function

$$\langle \ell_c(\theta; x, y, z) \rangle = \sum_n \langle \log p(z_n \mid x_n, \xi) \rangle_{p(z|x,y)} + \sum_n \langle \log p(y_n \mid x_n, z_n, \theta, \sigma) \rangle_{p(z|x,y)}$$

$$= \sum_n \sum_k \langle z_n^k \rangle \log\left(\mathrm{softmax}(\xi_k^T x_n)\right) - \frac{1}{2} \sum_n \sum_k \langle z_n^k \rangle \left(\frac{(y_n - \theta_k^T x_n)}{\sigma_k^2} + \log \sigma_k^2 + C \right)$$

- EM:
  - E-step: $\tau_n^{k(t)} = P(z_n^k = 1 | x_n, y_n, \theta) = \frac{p(z_n^k = 1 | x_n) p_k(y_n | x_n, \theta_k, \sigma_k^2)}{\sum_j p(z_n^j = 1 | x_n) p_j(y_n | x_n, \theta_j, \sigma_j^2)}$
  - M-step:
    - using the normal equation for standard LR $\theta = (X^T X)^{-1} X^T Y$, but with the data re-weighted by τ (homework)
    - IRLS and/or weighted IRLS algorithm to update $\{\xi_k, \theta_k, \sigma_k\}$ based on data pair (x_n, y_n), with weights $\tau_n^{k(t)}$ (homework)

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

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

% M-step

for each node i

$\theta_i := \text{MLE}(ESS_i)$

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.

EM Variants

- Sparse EM:

  Do not re-compute 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). Recall the IRLS step in the mixture of experts model

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