Latent variable models
Variational approximations.

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Variational methods

Variational methods have been used in applied math, physics, statistics, control theory, economics.

In this course:

• Used to support approximate inference and estimation when exact methods become hard
• An alternative to Monte Carlo method

Basic idea:
Assume a complex function of \( x \), \( f(x) \)
And a simpler function \( q \) with a set of parameters \( \lambda \), \( q(x, \lambda) \)
Approximate \( f(x) \) as:

\[
 f(x) \sim q(x, \lambda^*)
\]

where \( \lambda^* \) are parameters that yield the best approximation
Complex \( f(x) \) represented by a simpler \( q(x, \lambda) \) + optimization
Variational methods

Example:
• Assume a concave function $ln(x)$
• It can be upper bounded by a ‘simpler’ linear function
  $ln(x) \leq \lambda x - ln(\lambda) - 1$
• $\lambda$ - a variational parameter that helps to approximate $ln(x)$.

Variational methods: probabilistic inference

Support approximate inference and estimation

Example: Assume we have a probabilistic model with two sets of variables $x$ and $h$

(1) Assume we want to model $P(h | x)$ for all configurations of $h$
   If $h$ are binary then we need to model/calculate $2^k$ probabilities
(2) Assume we want to calculate $P(x) = \sum_{\{h\}} P(x, h)$
   It requires to sum $2^k$ terms
Cooperative vector quantizer

Latent variables (s):  binary vars
Dimensionality k

Observed variables x:  real valued vars
Dimensionality d

Model:
Latent var \( s_i \):
~ Bernoulli distribution
parameter: \( \pi_i \)
\[ P(s_i \mid \pi_i) = \pi_i^{s_i} (1 - \pi_i)^{1-s_i} \]

Observable variables x:
~ Normal distribution
parameters: \( W, \Sigma \)
\[ P(x \mid s) = N(Ws, \Sigma) \]
We assume \( \Sigma = \sigma I \)

Joint for one instance of x and s:
\[ P(x,s \mid \Theta) = \left(2\pi\right)^{-d/2} \sigma^{-d/2} \exp \left\{ -\frac{1}{2\sigma^2} (x - Ws)^T (x - Ws) \right\} \prod_{i=1}^{k} \pi_i^{s_i} (1 - \pi_i)^{1-s_i} \]
Cooperative vector quantizer

Our objective:
- **Learn the parameters of the model** $W, \pi, \sigma$
- **One can use the data likelihood or loglikelihood and optimize** ..

**Learning if $x$ and $s$ are observable**

Log likelihood:

$$
\sum_{n=1}^{N} \log P(x^{(n)}, s^{(n)} | \Theta) =
$$

$$
\sum_{n=1}^{N} -d \log \sigma - \frac{1}{2\sigma^2} (x^{(n)} - Ws^{(n)})^T (x^{(n)} - Ws^{(n)}) + \sum_{i=1}^{k} s_i^{(n)} \log \pi_i + (1 - s_i^{(n)}) \log (1 - \pi_i) + c
$$

Solution: nice and easy

---

Cooperative vector quantizer

Our objective:
- **Learn the parameters of the model** $W, \pi, \sigma$
- **One can use the data likelihood or loglikelihood and optimize** ..

**Learning if only $x$ are observable**

Log likelihood of data:

$$
\log P(D | \Theta) = \sum_{n=1}^{N} \log P(x^{(n)} | \Theta) = \sum_{n=1}^{N} \sum_{s^{(n)}} P(x^{(n)}, s^{(n)} | \Theta)
$$

Solution: does not let us benefit from the decomposition

EM: used to work in such cases …
EM

Let \( H \) – be a set of all variables with hidden or missing values
\[
P(H, D | \Theta, \xi) = P(H | D, \Theta, \xi)P(D | \Theta, \xi)
\]
\[
\log P(H, D | \Theta, \xi) = \log P(H | D, \Theta, \xi) + \log P(D | \Theta, \xi)
\]
\[
\log P(D | \Theta, \xi) = \log P(H, D | \Theta, \xi) - \log P(H | D, \Theta, \xi)
\]

\[
\text{Average both sides with } P(H | D, \Theta', \xi) \text{ for } \Theta'
\]
\[
E_{H|D,\Theta'} \log P(D | \Theta, \xi) = E_{H|D,\Theta'} \log P(H, D | \Theta, \xi) - E_{H|D,\Theta'} \log P(H | D, \Theta, \xi)
\]
\[
\log P(D | \Theta, \xi) = F(\Theta | \Theta') = E(\Theta | \Theta') + H(\Theta | \Theta')
\]

Log-likelihood of data

EM algorithm

Algorithm (general formulation)
Initialize parameters \( \Theta \)
Repeat
Set \( \Theta' = \Theta \)

1. Expectation step
\[
E(\Theta | \Theta') = \langle \log P(H, D | \Theta, \xi) \rangle_{P(H \mid D, \Theta')}
\]

2. Maximization step
\[
\Theta = \arg \max_{\Theta} E(\Theta | \Theta')
\]
until no or small improvement in \( \Theta \) (\( \Theta = \Theta' \))

Problem: posterior \( P(H \mid D, \Theta', \xi) \) is defined over \( 2^k \) probabilities
**EM algorithm**

Posterior $P(H \mid D, \Theta', \xi)$ for our model

$$P(H \mid D, \Theta') = \prod_{n=1}^{N} P(x^{(n)} \mid x^{(n)}, \Theta')$$

- Each data point $n=1, \ldots, N$ requires us to calculate $2^k$ probabilities.
- If $k$ is larger then this is a bottleneck!!!
**Variational approximation**

\[
E_{\|\lambda} \log P(D \mid \Theta, \xi) = E_{\|\lambda} \log P(H, D \mid \Theta, \xi) - E_{\|\lambda} \log Q(H \mid \lambda) \\
+ E_{\|\lambda} \log Q(H \mid \lambda) - E_{\|\lambda} \log P(H \mid \Theta, \xi)
\]

\[
\log P(D \mid \Theta, \xi) = F(Q, \Theta) + KL(Q, P)
\]

\[
F(Q, \Theta) = \sum_{\{H\}} Q(H \mid \lambda) \log P(H, D \mid \Theta, \xi) - \sum_{\{H\}} Q(H \mid \lambda) \log Q(H \mid \lambda)
\]

\[
KL(Q, P) = \sum_{\{H\}} Q(H \mid \lambda) \left[ \log Q(H \mid \lambda) - \log P(H \mid D, \Theta) \right]
\]

**Approximation:** jointly maximize \( F(Q, \Theta) \)

**Parameters:** \( \Theta, \lambda \)

**Why?** \( \log P(D \mid \Theta, \xi) \geq F(Q, \Theta) \)

Maximization of \( F \) pushes up the lower bound on the log-likelihood

---

**Variational approximation**

- **Comparison:**
  - EM uses true posterior \( P(H \mid D, \Theta', \xi) \)
  - Variational EM uses a surrogate posterior \( Q(H \mid \lambda) \)

**EM:**

\[
\log P(D \mid \Theta, \xi) = E_{\|\Theta, \xi} \log P(H, D \mid \Theta, \xi) - E_{\|\Theta, \xi} \log P(H \mid D, \Theta, \xi)
\]

\[
\log P(D \mid \Theta, \xi) = F(\Theta \mid \Theta')
\]

**Variational EM:**

\[
E_{\|\lambda} \log P(D \mid \Theta, \xi) = E_{\|\lambda} \log P(H, D \mid \Theta, \xi) - E_{\|\lambda} \log Q(H \mid \lambda) \\
+ E_{\|\lambda} \log Q(H \mid \lambda) - E_{\|\lambda} \log P(H \mid \Theta, \xi)
\]

\[
\log P(D \mid \Theta, \xi) = F(P, Q) + KL(Q, P)
\]

\[
\log P(D \mid \Theta, \xi) \geq F(P, Q)
\]
Variational EM

Let $H$ – be a set of all variables with hidden or missing values

- **E step:**
  - Optimize $F(Q, \Theta)$ with respect to $\lambda$ while keeping $\Theta$ fixed

- **M step**
  - Optimize $F(Q, \Theta)$ with respect to $\Theta$ while keeping $\lambda$s

Note: if $Q(H)$ is the posterior then the variational EM reduces to the standard EM

---

Variational EM

- So what is the deal?
  - Why should we use the variational EM?
- Hope:
  - If we choose $Q(H | \lambda)$ well the optimization of both $\lambda$ and $\Theta$ will become easy !!!!
- A well behaved choice for $Q(H | \lambda)$
  - the mean field approximation

$$Q(H | \lambda) = \prod_i Q_i(H_i | \lambda_i)$$
Mean Field Approximation

Assumption:
- $Q(H | \lambda)$ is the mean field approximation.
- Variables in the $Q(H)$ distribution are independent variables $H_i$.
- $Q$ is completely factorized:
  - For our CVQ model
  - Hidden variables are binary sources
    
$Q(H) = \prod_{i=1}^{N} Q(s_i^{(n)} | \lambda_i^{(n)})$

$Q(s_i^{(n)} | \lambda_i^{(n)}) = \prod_{i=1}^{d} Q(s_i^{(n)} | \lambda_i^{(n)})$

$Q(s_i^{(n)} | \lambda_i^{(n)}) = \lambda_i^{(n)} s_i^{(n)} (1- \lambda_i^{(n)}) (1-s_i^{(n)})$

Mean Field Approximation

Functional $F$ for the mean field:

$F(Q, \Theta) = \sum_{H} Q(H | \lambda) \log P(H, D | \Theta, \xi) - \sum_{H} Q(H | \lambda) \log Q(H | \lambda)$

Assume just one data point $x$ and corresponding $s$ :

$F(Q, \Theta) = \langle \log P(x, s | \Theta) \rangle_{Q(\lambda | \xi)} - \langle \log Q(s | \lambda) \rangle_{Q(\lambda | \xi)}$

$= \left\langle -d \log \sigma - \frac{1}{2 \sigma^2} (x - Ws)^T (x - Ws) \right\rangle_{Q(\lambda | \xi)}$

$+ \left\langle \sum_{i=1}^{k} s_i \log \pi_i + (1-s_i) \log(1-\pi_i) \right\rangle_{Q(\lambda | \xi)}$

$- \left\langle \sum_{i=1}^{k} s_i \log \lambda_i + (1-s_i) \log(1-\lambda_i) \right\rangle_{Q(\lambda | \xi)}$
Mean Field Approximation

Functional F. Part 1:

\[ \left\langle -d \log \sigma - \frac{1}{2\sigma^2} (x - \sum_{i=1}^{k} s_i w_i)^T (x - \sum_{i=1}^{k} s_i w_i) \right\rangle_{Q(\cdot|\cdot)} = \]

\[ = \left\langle -d \log \sigma - \frac{1}{2\sigma^2} (x - \sum_{i=1}^{k} s_i w_i)^T (x - \sum_{i=1}^{k} s_i w_i) \right\rangle_{Q(\cdot|\cdot)} \]

\[ = -d \log \sigma - \frac{1}{2\sigma^2} \left[ x^T x - 2 \sum_{i=1}^{k} \langle s_i \rangle_{Q(s_i|\cdot)} w_i x + \sum_{i=1}^{k} \sum_{j=1}^{k} \langle s_i s_j \rangle_{Q(s_i, s_j|\cdot)} w_i w_j \right]_{Q(\cdot|\cdot)} \]

\[ \langle s_i \rangle_{Q(s_i|\cdot)} = \lambda_i \]

\[ \langle s_i s_j \rangle_{Q(s_i, s_j|\cdot)} = \lambda_i \lambda_j + \delta_{ij} (\lambda_i - \lambda_i^2) \]

---

Mean Field Approximation

Functional F. Part 2:

\[ \left\langle \sum_{i=1}^{k} s_i \log \pi_i + (1 - s_i) \log(1 - \pi_i) \right\rangle_{Q(s_i|\cdot)} = \sum_{i=1}^{k} \langle s_i \rangle_{Q(s_i|\cdot)} \log \pi_i + (1 - \langle s_i \rangle_{Q(s_i|\cdot)}) \log(1 - \pi_i) \]

\[ = \sum_{i=1}^{k} \lambda_i \log \pi_i + (1 - \lambda_i) \log(1 - \pi_i) \]

Functional F. Part 3:

\[ \left\langle \sum_{i=1}^{k} s_i \log \lambda_i + (1 - s_i) \log(1 - \lambda_i) \right\rangle_{Q(s_i|\cdot)} = \sum_{i=1}^{k} \lambda_i \log \lambda_i + (1 - \lambda_i) \log(1 - \lambda_i) \]
Mean Field Approximation

Functional F:
\[ F(Q, \Theta) = \langle \log P(x, s) | \Theta \rangle_{Q(x,s)} - \langle \log Q(s | \lambda) \rangle_{Q(s | \lambda)} \]
\[ = -d \log \sigma - \frac{1}{2\sigma^2} \left[ x^T x - 2 \sum_{i=1}^{k} \lambda_i w_i x + \sum_{i=1}^{k} \sum_{j=1}^{k} \left[ \lambda_i \lambda_j + \delta_{ij} (\lambda_i - \lambda_i^2) \right] w_i^T w_j \right] \]
\[ + \sum_{i=1}^{k} \lambda_i \log \pi_i + (1 - \lambda_i) \log(1 - \pi_i) \]
\[ + \sum_{i=1}^{k} \lambda_i \log \lambda_i + (1 - \lambda_i) \log(1 - \lambda_i) \]

Parameters: \( W, \pi, \sigma \)
Mean field parameters: \( \lambda \)

Mean Field Approximation

Functional F (for all data points):
\[ F(Q, \Theta) = \sum_{n=1}^{N} \left[ \log P(x^{(n)}, s^{(n)} | \Theta) \right]_{Q(x^{(n)}, s^{(n)})} - \left[ \log Q(s^{(n)} | \lambda^{(n)}) \right]_{Q(s^{(n)} | \lambda^{(n)})} \]
\[ = -d \log \sigma - \frac{1}{2\sigma^2} \left[ x^{(n)^T} x^{(n)} - 2 \sum_{i=1}^{k} \lambda_i^{(n)} w_i x^{(n)} + \sum_{i=1}^{k} \sum_{j=1}^{k} \left[ \lambda_i^{(n)} \lambda_j^{(n)} + \delta_{ij} (\lambda_i^{(n)} - \lambda_i^{(n)^2}) \right] w_i^{(n)^T} w_j^{(n)} \right] \]
\[ + \sum_{i=1}^{k} \lambda_i^{(n)} \log \pi_i + (1 - \lambda_i^{(n)}) \log(1 - \pi_i) \]
\[ + \sum_{i=1}^{k} \lambda_i^{(n)} \log \lambda_i^{(n)} + (1 - \lambda_i^{(n)}) \log(1 - \lambda_i^{(n)}) \]

Parameters: \( W, \pi, \sigma \)
Mean field parameters: \( \lambda = \lambda^{(1)}, \lambda^{(2)}, \ldots \lambda^{(N)} \)
Variational EM: E step

Optimization of the functional $F$ with respect to $\lambda$:

$$\frac{\partial}{\partial \lambda_u} F = \frac{1}{\sigma^2} (x - \sum_{j \neq u} \lambda_j w_j)^T w_u - \frac{1}{2\sigma^2} w_u^T w_u + \log \frac{\pi_u}{1-\pi_u} - \log \frac{\lambda_u}{1-\lambda_u}$$

Set $\frac{\partial}{\partial \lambda_u} F = 0$

$$\lambda_u = g \left( \frac{1}{\sigma^2} (x - \sum_{j \neq u} \lambda_j w_j)^T w_u - \frac{1}{2\sigma^2} w_u^T w_u + \log \frac{\pi_u}{1-\pi_u} \right)$$

$g(x) = \frac{1}{1+e^{-x}}$

Defines a fixed point equation

Iterate a set fixed point equations for all indexes $u=1..k$ and for all $n$.

Variational EM: M step

Optimization of the functional $F$ with respect to $\Theta$. Start with $\pi$:

For $N$ data points

$$\frac{\partial}{\partial \pi_u} F = \sum_{n=1}^{N} \lambda_u^n \log \frac{1}{\pi_u} - (1-\lambda_u^n) \log \frac{1}{1-\pi_u}$$

Set $\frac{\partial}{\partial \pi_u} F = 0$

$$\pi_u = \frac{\sum_{n=1}^{N} \lambda_u^{(n)}}{N} \quad \text{Closed form solution}$$
Variational EM: M step

Optimization of the functional $F$ with respect to $\Theta$.

Parameters $w$:

$$\frac{\partial}{\partial w_{uv}} F = \sum_{n=1}^{N} \frac{1}{2\sigma^2} \left[ \lambda_v^{(n)} x_u^{(n)} + 2\sum_{j \neq v} \lambda_j^{(n)} w_{uj} + 2\lambda_v^{(n)} w_{uv} \right] = 0$$

$$W = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1k} \\ w_{21} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & w_{dk} \end{pmatrix}$$

$W = (w_1 \ w_2 \ \ldots \ w_k)$

For each variable $v$:

The equations define a set of $k$ linear equations that can be solved

---

Bayesian CVQ Model

For each variable $v$:

The equations define a set of $k$ linear equations that can be solved

---

Bayesian model:

Distributions over parameters

$$y = \sum_{k=1}^{K} s_k w_k + \mathcal{E}$$

$$P(y \mid S, \theta) \sim N \left( \sum_{k=1}^{K} s_k w_k, \tau^{-1}I \right)$$

---

CS 2750 Machine Learning
**Model Specification**

\[ X = \{x_1, x_2, \ldots, x_n\} \quad \text{observed data} \]

\[ S = \{s_1, \ldots, s_k\} \quad \text{latent sources} \]

\[ \pi = \{\pi_1, \pi_2, \ldots, \pi_k\} \quad \text{probability of } s_k = 1 \]

\[ W = \{w_1, w_2, \ldots, w_k\} \quad \text{DxK weight matrix} \]

\[ \gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_k\} \quad \text{Variance of } W \]

\[ \tau \quad \text{Precision of noise} \]

---

**Priors**

\[ P(\pi) = \prod_{k=1}^{K} Beta(\pi_k \mid \alpha, \beta) \]

\[ P(W) = \prod_{k=1}^{K} N(w_k \mid 0, \gamma_k) \]

\[ P(\gamma) = \prod_{k=1}^{K} Gamma(\gamma_k \mid a_{\gamma}, b_{\gamma}) \]

\[ P(\tau) = Gamma(\tau \mid c_{\tau}, d_{\tau}) \]
Why the Bayesian model?

Very useful for Bayesian Model Selection
- Assume we do not know the number of sources
- Bayesian score tells us how good the structure is

Benefits of the Bayesian score:
- Embodies Occam’s Razor
- Prevents overfit

\[
P(M_i | D) = \frac{P(D | M_i) P(M_i)}{P(D)}
\]

\[
P(D | M_i) = \int_{\theta} P(D | \theta, M_i) P(\theta | M_i)
\]

- Marginal likelihood

Variational approximation

- Approximation: loglikelihood of data

\[
\log P(X) = \log \int_{\theta} P(X, \theta) d\theta
\]

\[
= \log \int_{\theta} \sum_{H} P(X, H, \theta) d\theta
\]

\[
= \log \int_{\theta} \sum_{H} P(X, H | \theta) P(\theta) d\theta
\]

\[
\geq \int_{\theta} \sum_{H} Q(H, \theta) \log \frac{P(X, H | \theta) P(\theta)}{Q(H, \theta)} d\theta = F(Q)
\]

Where Q is a distribution with different parameterization
**Variational approximation**

- Approximation: loglikelihood of observable data
  \[
  \log P(X) = F(Q) + KL(Q(H, \theta), P(H, \theta))
  \]
- Optimization of F(Q) is pushing up the lower bound on the loglikelihood of observable data
- How to choose Q?
  \[
  Q(H, \theta) = Q_{\theta}(\theta)Q_H(H)
  \]
- Then:
  \[
  F(Q) = \int_{\theta} Q_{\theta}(\theta) \left[ \sum_H Q_H(H) \log \frac{P(X, H | \theta)}{Q_H(H)} \right] d\theta
  \]
  \[
  + \int_{\theta} Q_{\theta}(\theta) \log \frac{Q_{\theta}(\theta)}{P(\theta)} d\theta \quad \text{← KL distance}
  \]

**Variational Bayes approximation**

- Evaluation of Q(H, \theta) is intractable
- Meanfield approximation
  \[
  Q(H, \theta) = \prod_{k=1}^{K} Q(H_k) \prod_{i=1}^{P} (\theta_i)
  \]
- Allows analytical evaluation of F(Q)
**VB learning**

Learn Model with an EM like algorithm

(1) VBE – Optimize $Q(H)$

Estimate state of latent variables

$$Q^*_H(H) \propto \exp \left\{ \log P(D, H \mid \theta) \right\}_{Q_\theta(\theta)}$$

(2) VBM – Optimize $Q(\Theta)$

Estimate parameters

$$Q^*_\theta(\theta) \propto P(\theta) \exp \left\{ \log P(D, H \mid \theta) \right\}_{Q_H(H)}$$

See Ghahramani & Beal 2004 for the details
VBM

\[ Q_\pi (\pi) = \prod_{k=1}^{K} \text{Beta}(\pi_k \mid \tilde{\alpha}_k, \tilde{\beta}_k) \]

\[ Q_W (W) = \prod_{d=1}^{D} \mathcal{N}(w_k \mid \tilde{m}_w^{(d)}, \tilde{\Sigma}_w^{(d)}) \]

\[ Q_\gamma (\gamma) = \prod_{k=1}^{K} \text{Gamma}(\gamma_k \mid \tilde{\alpha}_{jk}, \tilde{\beta}_{jk}) \]

\[ Q_\tau (\tau) = \text{Gamma}(\tau \mid \tilde{c}_\tau, \tilde{d}_\tau) \]

VBM (cont’)

\[ \tilde{\alpha}_k = \alpha_k + \sum_{n=1}^{N} s_k^{(n)} \]

\[ \tilde{\beta}_k = \beta_k + N - \sum_{n=1}^{N} s_k^{(n)} \]

\[ \tilde{\Sigma}_w^{(d)} = \left( \text{diag}(s) + \langle \tau \rangle \sum_{n=1}^{N} s_n s_n^T \right)^{-1} \]

\[ \tilde{m}_w^{(d)} = \tilde{\Sigma}_w^{(d)} \langle \tau \rangle \sum_{n=1}^{N} s_n y_d^{(n)} \]
VBM (cont’)

\[ \tilde{a}_{jk} = a_{jk} + \frac{D}{2} \]

\[ \tilde{b}_{jk} = b_{jk} + \frac{\|w_k\|^2}{2} \]

\[ \tilde{c}_r = c_r + \frac{ND}{2} \]

\[ \tilde{d}_r = d_r + \frac{1}{2} \sum_{n=1}^{N} \left\{ \|y^n\|^2 - 2y^{nT} \langle W \rangle s^n \right\} \]

\[ + \text{tr} \left( \langle W^T W \rangle \langle s^n s^{nT} \rangle \right) \]

---

Image Separation Experiment

- 8 sources images projected to \( x \)
- A binary switch defines whether the image is projected or not
Mixed images

Recovered sources

Initial sources

Recovered sources
The Noisy-OR Vector Quantizer

Tomas Singliar and Milos Hauskrecht

Outline

• High-dimensional data models
• Latent Variable Models
• Noisy-or model (for binary data)
• Learning LVM with EM
• The approximate learning algorithm
• Experiments: Data mining with Noisy-OR VQ
• Conclusions
Modeling high-dimensional data

- High-dimensional data
  - sensor networks
  - document repositories
- typically variables are dependent
- How to model dependencies?
  - Full model intractable, overfitting
  - All-independent unrealistic
  - Middle-of-the-road models
    - capture dependencies
    - efficient – representation, reasoning, learning

Latent Factor Model

- Captures dependencies via latent factors and their combinations

\[ \begin{align*}
K \text{ nodes } s & \quad \quad \text{hidden layer} \\
D \text{ nodes } x & \quad \quad \text{observed layer} \\
\end{align*} \]

- \( K \ll D \): Dimensionality reduction, data compression

- Binary hidden layer – vector quantization
Compact Parameterization

- CPT specifies $P(X|\text{parents}(X))$
- full model would have $2^K$ entries
- Binary nodes – Noisy-OR parameterization
  - $K$ parameters for each observed node $p_{ij}$, ..., $p_{kj}$
  - $p_{ij}$ “strength of influence” of $s_i$ on observable $x_j$

$$P(X_j = 0 | s) = \prod_{i=1}^{K} (1 - p_{ij})^{s_i}$$
$$P(X_j = 1 | s) = 1 - P(X_j = 0 | s) = 1 - \prod_{i=1}^{K} (1 - p_{ij})^{s_i}$$

Exponential Reparameterization

- Exponential form easier to work with
- New parameters $\theta_{ij} = -\log(1 - p_{ij})$

$$P(X_j = 0 | s) = \prod_{i=1}^{K} (1 - p_{ij})^{\theta_{ij}} = \prod_{i=1}^{K} \exp(-\theta_{ij}s_i)$$
$$P(X_j = 1 | s) = 1 - P(X_j = 0 | s) = 1 - \prod_{i=1}^{K} \exp(-\theta_{ij}s_i)$$
$$= 1 - \exp\left[-\sum_{i} \theta_{ij}s_i\right] = \exp\left(\log\left[1 - \exp\left(-\sum_{i} \theta_{ij}s_i\right)\right]\right)$$
Learning with hidden variables - EM

- Expectation – maximization: standard MLL technique
  - E-step: Compute $p(H \mid D, \theta')$ - fill in the data
  - M-step: Choose new $\theta$ to maximize complete LL:
    $$Q(\theta \mid \theta') = \langle \log P(D, H \mid \theta) \rangle_{P(H \mid D, \theta')}$$
    - $\Theta' \leftarrow \Theta$ and repeat until convergence
- Guaranteed to improve $Q$ every iteration
- Local optimization method
- Following EM, we want to optimize
  $$\Theta = \arg \max \langle \log P(x, s \mid \Theta) \rangle_{P(s \mid x, \Theta)}$$

Why EM won’t work

- N iid samples (D-dimensional binary vectors)
- We will need:
  - The joint distribution
    $$P(x, s) = P(x \mid s) P(s) = P(s) \prod_j \left(1 - \prod_{i=1}^{K} (1 - p_{ij})^{s_i} \right)^{x_j} \left(\prod_{i=1}^{K} (1 - p_{ij})^{s_i} \right)^{1-x_j}$$
  - Joint over observables
    $$P(x) = \sum_s P(x, s) = \sum_s \left( \prod_j P(x_j \mid s) \right) P(s)$$
    - **Problem 1: not a product**
    - **Problem 2: summation over $2^k$ terms**
**Decomposition**

If we can express $P(x_j | s)$ as $P(x_j | s) = \prod_i h(x_j | s_i)$

$$P(x) = \sum_s \left( \prod_j P(x_j | s) \left( \prod_i P(s_i) \right) \right)$$

$$= \sum_s \left( \prod_j \prod_i h(x_j | s_i) \left( \prod_i P(s_i) \right) \right)$$

$$= \prod_i P(s_i) \sum_s \prod_j h(x_j | s_i)$$

- joint over observables decomposes along hidden factors
- how do we find $h(x_j | s_i)$?

---

**Decomposable lower bound**

- Based on Jensen's inequality

$$P(X_j = 1 | s) = \exp \left( \log \left( 1 - \exp \left[ -\sum_i \theta_i s_i \right] \right) \right)$$

$$\geq \exp \left( \sum_i q_j(i)s_i \log \left( 1 - e^{-\theta_i/q_j(i)} \right) + C \right) = \prod_i h_L(x_j | s_i)$$

This is a product! Solves Problem 1

Variational Parameters satisfying $\sum_j q_j = 1$
Inference on latent sources

- The loglikelihood bound decomposes (Problem 1 solved)
- We can now compute posteriors efficiently

\[ \tilde{P}(s_i = 1 \mid \mathbf{x}) = \frac{P(s_i = 1) \prod_j h_L(x_j \mid s_i = 1)}{\sum_{s_i} P(s_i) \prod_j h_L(x_j \mid s_i)} \]  
Solves Problem 2

- \(O(D)\) operations required to perform inference

EM vs Variational EM

**Classical EM**
- Iterate until convergence of \(F:\)
  - Compute expectation over hidden variables (E-step)
  - Maximize expected loglikelihood (M-step)
    - optimize w.r.t. \(\Theta\) (model parameters)

As if we had additional model parameters

**Variational EM**
- Iterate until convergence of \(F:\)
  - Compute expectation over hidden variables (E-step)
  - Maximize expected loglikelihood (M-step)
    - optimize w.r.t. \(q\) (variational parameters)
    - a fixed-point equation
    - optimize w.r.t. \(\Theta\) (model parameters)
      - numerical search
Structure recovery experiments

- Similar source “Source separation” task
- Synthetic data: small B/W pictures
  - true scramble + noise recover

Real-world data mining with Noisy-OR VQ

- CiteSeer data: 40 ML authors, ~5000 papers
- Create dataset:
  - $M(i,j) = 1$ if document $i$ cites author $j$, 0 otherwise (5000x40 mx)
- Run learning and see components:
  - 1: Friedman, Jordan, Lauritzen, Pearl
  - 3: Scholkopf, Smola, Vapnik, Burges
  - 4: Ghahramani, Jordan, Hintom, Neal, Saul, Bishop, Tipping
  - 5: Frey, Freeman, Murphy, Lauritzen, Pearl, Weiss, Yedidia
- Components reflect ML communities