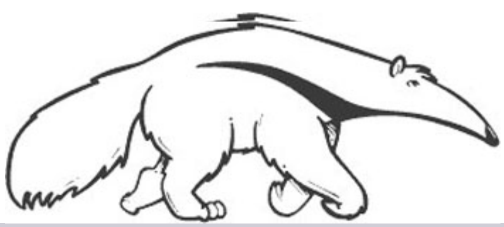
Latent Variable Models

Learning in Graphical Models

Prof. Alexander Ihler



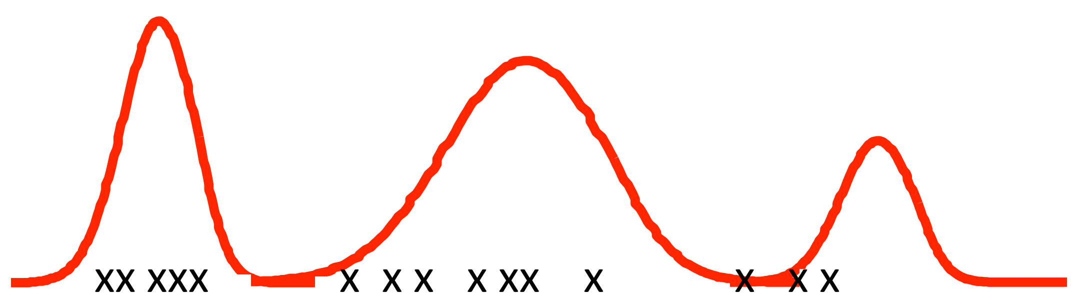
BREN:ICS INFORMATION AND COMPUTER SCIENCES

UNIVERSITY of CALIFORNIA IRVINE

Latent variable models

- Never observe some variables?
- Ex: Gaussian mixture models

- Probability distribution: $p(x) = \sum_c \pi_c \, \mathcal{N}(x \; ; \; \mu_c, \sigma_c)$



Latent variable models

- Never observe some variables?
- Ex: Gaussian mixture models

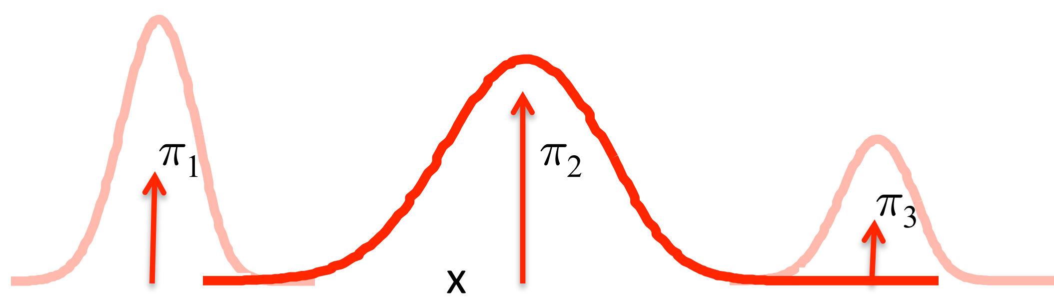
- Probability distribution: $p(x) = \sum_c \pi_c \, \mathcal{N}(x \; ; \; \mu_c, \sigma_c)$
- Equivalent "latent variable" form:

$p(z = c) = \pi_c$ Select a mixture component with probability π

$p(x | z = c) = \mathcal{N}(x \; ; \; \mu_c, \sigma_c)$ Sample from that component's Gaussian

"Latent assignment" z:
we observe x, but z is hidden

p(x) = marginal over x



Learning mixture models

- Maximum likelihood? $p(x) = \sum_c \pi_c \, \mathcal{N}(x \; ; \; \mu_c, \sigma_c)$

- Observe iid samples D = {$x^{(1)}$ … $x^{(m)}$}

$$\mathcal{L}_X(\theta = \{\pi, \mu, \sigma^2\}) = \sum_j \log \left[\sum_c \pi_c h(\sigma_c) \exp((x^{(j)} - \mu_c)^2 / \sigma_c^2) \right]$$

- Gradient descent?

$$\frac{\partial \mathcal{L}_X(\theta)}{\partial \pi_c} = \dots \qquad \frac{\partial \mathcal{L}_X(\theta)}{\partial \mu_c} = \dots \qquad \frac{\partial \mathcal{L}_X(\theta)}{\partial \sigma_c} = \dots$$

- This can be very slow…

Recall ML for Bayes Nets

$$x_1 \rightarrow x_2 \rightarrow x_3$$

- Fully observed:

$$\mathcal{L} = \sum_i \log \left[p(x_1^i)\, p(x_2^i | x_1^i)\, p(x_3^i | x_2^i) \right]$$

Likelihood decomposes; closed form MLE

$$D = \begin{bmatrix} x^1 \\ x^2 \\ x^3 \\ x^4 \\ \vdots \end{bmatrix} = \begin{array}{c} \begin{array}{ccc} x_1 & x_2 & x_3 \end{array} \\ \begin{bmatrix} x_1^1 & x_2^1 & x_3^1 \\ x_1^2 & x_2^2 & x_3^2 \\ x_1^3 & x_2^3 & x_3^3 \\ x_1^4 & x_2^4 & x_3^4 \\ \vdots & \vdots & \vdots \end{bmatrix} \end{array}$$

What if...

- Not fully observed?

$$x_1 \rightarrow x_2 \rightarrow x_3$$

$$\mathcal{L} = \sum_i \log \left[p(x_1^i)\, p(x_2^i | x_1^i)\, p(x_3^i | x_2^i) \right]$$

$$D = \begin{bmatrix} x^1 \\ x^2 \\ x^3 \\ x^4 \\ \vdots \end{bmatrix} = \begin{array}{c} \begin{array}{ccc} x_1 & x_2 & x_3 \end{array} \\ \begin{bmatrix} x_1^1 & ? & x_3^1 \\ x_1^2 & x_2^2 & x_3^2 \\ ? & x_2^3 & x_3^3 \\ x_1^4 & ? & ? \\ \vdots & \vdots & \vdots \end{bmatrix} \end{array}$$

$$\mathcal{L} = \log \left[p(x_1^1)\, p(?|x_1^1)\, p(x_3^1|x_2^1) \right] + \ldots$$

What if…

- Not fully observed?

$$x_1 \rightarrow x_2 \rightarrow x_3$$

$$\mathcal{L} = \sum_i \log \left[p(x_1^i)\, p(x_2^i | x_1^i)\, p(x_3^i | x_2^i) \right]$$

$$D = \begin{bmatrix} x^1 \\ x^2 \\ x^3 \\ x^4 \\ \vdots \end{bmatrix} = \begin{array}{c} \begin{array}{ccc} x_1 & x_2 & x_3 \end{array} \\ \begin{bmatrix} x_1^1 & ? & x_3^1 \\ x_1^2 & x_2^2 & x_3^2 \\ ? & x_2^3 & x_3^3 \\ x_1^4 & ? & ? \\ \vdots & \vdots & \vdots \end{bmatrix} \end{array}$$

$$\begin{aligned} \mathcal{L} &= \log \left[p(x_1^1)\, p(?|x_1^1)\, p(x_3^1|x_2^1) \right] + \dots \\ &= \log \left[\sum_{x_2} p(x_1^1)\, p(x_2|x_1^1)\, p(x_3^1|x_2) \right] + \dots \\ &= \sum_i \log \sum_{x_j \in S^i} p(x^i) \end{aligned}$$

No longer decomposes nicely into individual parts…
Hard to estimate in closed form.

Solutions?

1. Optimize directly anyway (gradient ascent, etc.)
2. Discard data with missing entries
3. Fill in ("impute") missing entries somehow

Complete log-likelihood

- Gaussian mixture model, log-likelihood

$$\mathcal{L}_X(\theta = \{\pi, \mu, \sigma^2\}) = \sum_j \log \left[\sum_k \pi_k h(\sigma) \exp((x^{(j)} - \mu)^2/\sigma^2) \right]$$

- If we observed z: the "complete data likelihood"

$$\mathcal{L}_{XZ}(\theta) = \sum_j \log p(z^{(j)}) p(x^{(j)} | z^{(j)})$$

$$= \sum_j \log \pi_{z^{(j)}} + (x^{(j)} - \mu_{z^{(j)}})^2/\sigma^2_{z^{(j)}} + \ldots$$

- Now, given z, this is exponential family – easy:

$$\hat{\pi}_k = \frac{1}{m} \sum \delta(z^{(j)} = k) = \frac{m_k}{m} \qquad \hat{\mu}_k = \frac{1}{m_k} \sum_{z^j = k} x^{(j)} \qquad \ldots$$

Learning with hidden data

- "Estimate" z, then optimize complete LL

- Example: *k-means-like* algorithm ("hard EM")
  - Find "best" z-values, then best means; repeat

- Expectation-Maximization
  - Instead of fixing z's, take "soft assignment"
  - Use q(z=1) … q(z=k)   (as if "partial" observations)

Expectation-Maximization

- Recall the K-L divergence between q & p:

$$D\big(\hat{q}(x) \| p(x)\big) = \mathbb{E}_q\big[\log \frac{q(x)}{p(x)}\big]$$

- D(q||p) $\geq$ 0, = 0 iff p=q a.e.

- Also the empirical distribution $\hat{p}(x) = \frac{1}{m}\sum_j \delta(x = x^{(j)})$

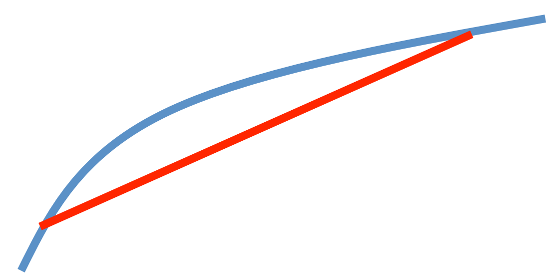
$$D\big(\hat{p}(x) \,\|\, p(x\,;\,\theta)\big) = \frac{1}{m}\sum_j \delta(x = x^{(j)})\big(\log \hat{p}(x) - \log p(x\,;\,\theta)\big)$$

$$= H(\hat{p}) - \frac{1}{m}\mathcal{L}(\theta)$$

Expectation-Maximization

$$-\frac{1}{m}\mathcal{L}_X + \hat{H}_X$$

$$\begin{aligned}
D\big(\hat{p} \| p(x\,;\,\theta)\big) &= -\mathbb{E}_{\hat{p}} \log p(x\,;\,\theta) - H(\hat{p}) \\
&= -\mathbb{E}_{\hat{p}} \big[\log \sum_z p(x, z\,;\,\theta)\big] - H(\hat{p}) \\
&= -\mathbb{E}_{\hat{p}} \big[\log \sum_z q(z|x) \frac{p(x, z\,;\,\theta)}{q(z|x)}\big] - H(\hat{p}) \\
&= -\mathbb{E}_{\hat{p}} \Big[\log \mathbb{E}_q \big[\frac{p(x, z\,;\,\theta)}{q(z|x)}\big]\Big] - H(\hat{p}) \\
&\le -\mathbb{E}_{\hat{p}} \mathbb{E}_q \big[\log \frac{p(x, z\,;\,\theta)}{q(z|x)}\big] - H(\hat{p}) \qquad \textit{(Jensen's ineq.)} \\
&= D\big(\hat{p}(x) \cdot q(z|x) \| p(x, z\,;\,\theta)\big)
\end{aligned}$$



Expectation-Maximization

$$\max_{\theta} \max_{q} \; -D\big(\hat{p}(x)q(z|x) \| p(x,z\,;\,\theta)\big)$$

- E-step: max with respect to q
  - Optimal q(z|x) = p(z | x, θ)

$$\mathbb{E}_{\hat{p}} \; \mathbb{E}_q \Big[\log \frac{p(x|\theta)p(z|x,\theta)}{\hat{p}(x)q(z|x)} \Big] \xrightarrow{q=p(z|x,\theta)} \mathbb{E}_{\hat{p}} \Big[\log \frac{p(x|\theta)}{\hat{p}(x)} \Big] \quad = \frac{1}{m}\mathcal{L}_X$$

- M-step: max with respect to θ
  - Given weights of q(z|x)

$$\mathbb{E}_q \; \mathbb{E}_{\hat{p}} \Big[\log \frac{p(x\,;\,\theta)p(z|x\,;\,\theta)}{\hat{p}(x)q(z|x)} \Big] = \mathbb{E}_q \; \mathbb{E}_{\hat{p}} \Big[\log p(z\,;\,\theta)p(x|z\,;\,\theta) \Big] + \text{ const.}$$

(Expected Complete Log-Likelihood)

Expectation-Maximization

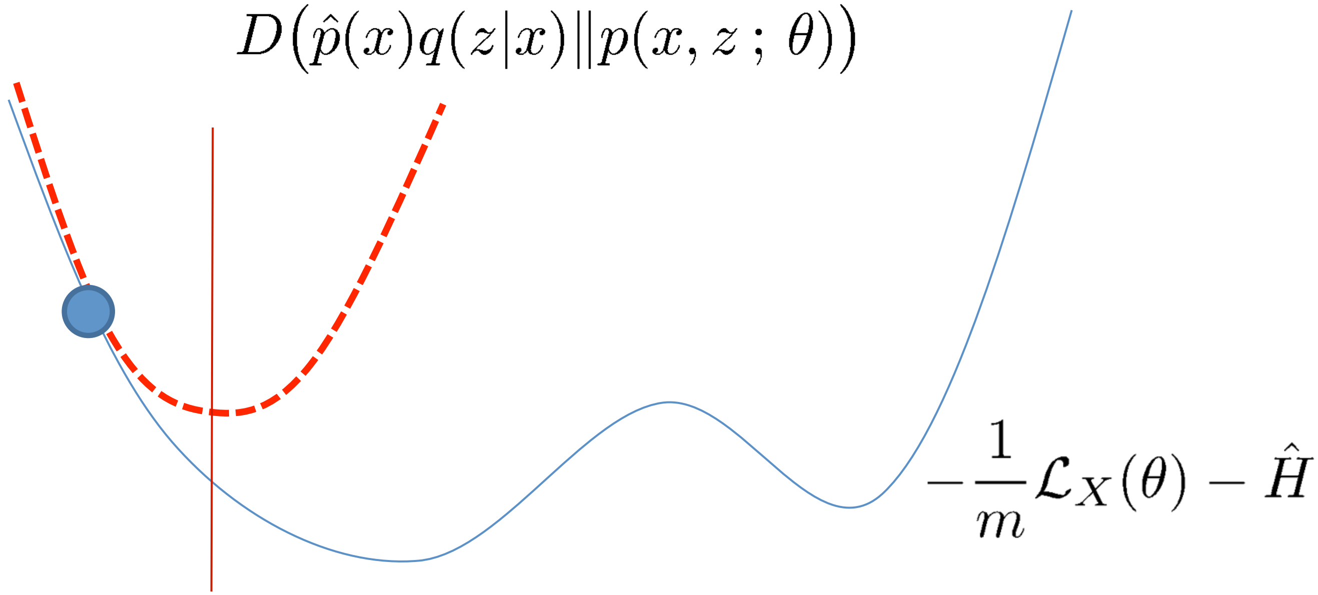
$$\max_\theta \max_q \; -D\big(\hat{p}(x)q(z|x) \| p(x, z\,;\,\theta)\big)$$

- E-step: max with respect to q
  - Optimal q(z) = p(z | x, θ)
- M-step: max with respect to θ
  - Expected complete log-likelihood

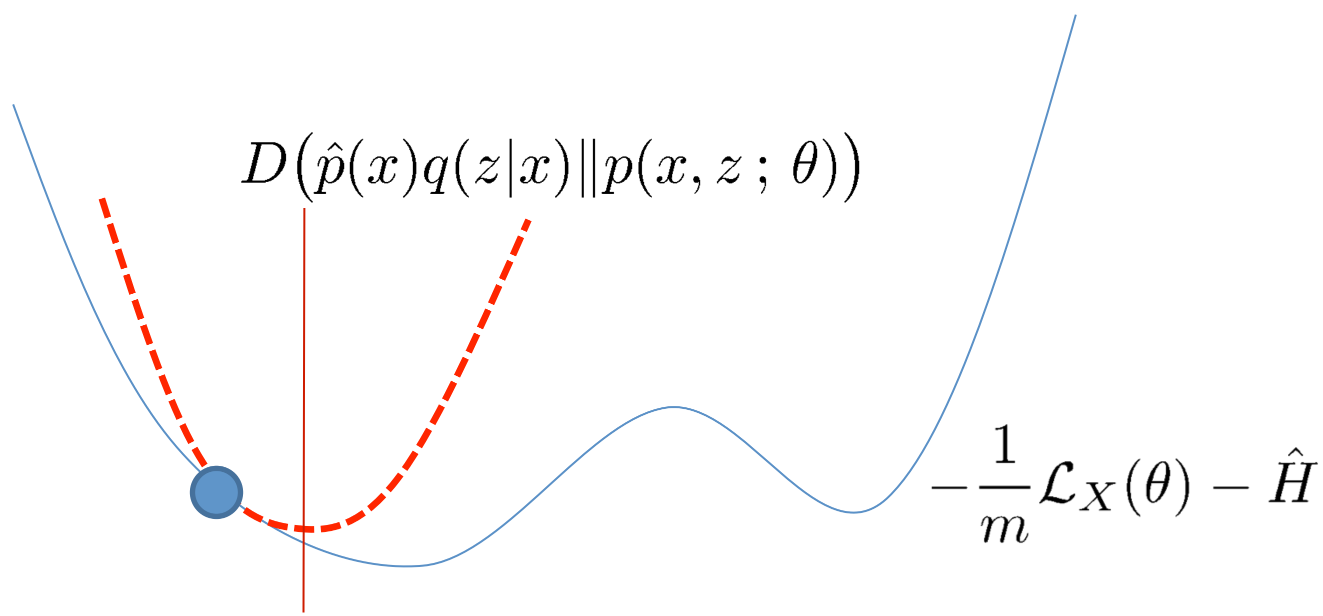
- Coordinate ascent
  - Always converges; maximizes lower bound on LL
  - Touches (equality) after E-step

Note: if we restrict q(z) to 0/1, this is *hard EM* (k-means-like)

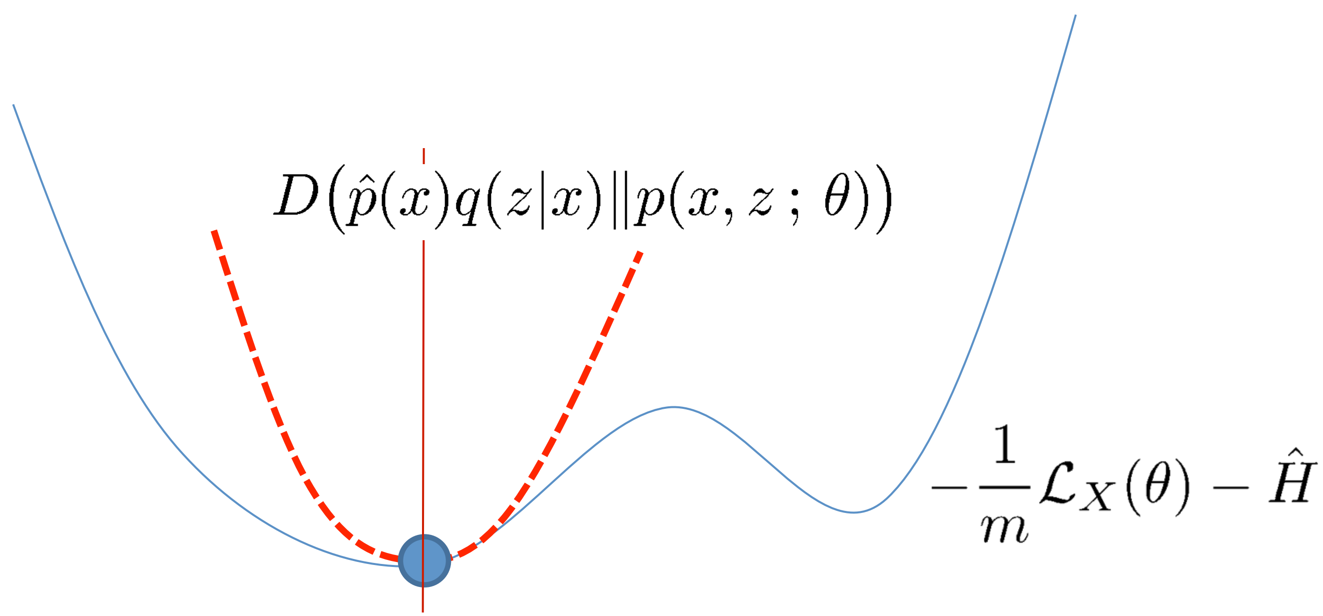
Expectation-Maximization



Expectation-Maximization



Expectation-Maximization



Expectation-Maximization

- Intuition:
  - Instead of fixing z's, take "soft assignment"
  - Use q(z=1) … q(z=k)   (as if "partial" observations)

- Derivation: optimize the expected complete LL

$$\max_{\theta} \mathbb{E}_{q(Z)}\left[\mathcal{L}(X,Z)\right] = \sum_{Z} q(Z)\underline{\mathcal{L}(X,Z)}$$

= log p(x,z)
p(x,z) is structured (factors into a product)
=> log p decomposes into a sum of log terms

Optimizing the ECLL

- Usually not harder than standard ML estimation for the CLL

$$\max_\theta \mathbb{E}_{q(Z)}\left[\mathcal{L}_{XZ}(\theta)\right] = \sum_Z q(Z)\mathcal{L}_{XZ}(\theta)$$

$$\frac{\partial}{\partial\theta_i}\mathcal{L}_{\mathcal{X}\mathcal{Z}}(\theta) = \sum_j \sum_k \underbrace{q(z^{(j)} = k)}_{\text{Weight, Wjk}} \frac{\partial}{\partial\theta_i}\left[\log \pi_k - \frac{1}{2}(x^{(j)} - \mu_k)^2/\sigma_k^2\right]$$

π, μ, σ

Weight, Wjk

$$u_i(x^{(j)}, z = k) - \mathbb{E}[u_i(x^{(j)}, z = k)]$$

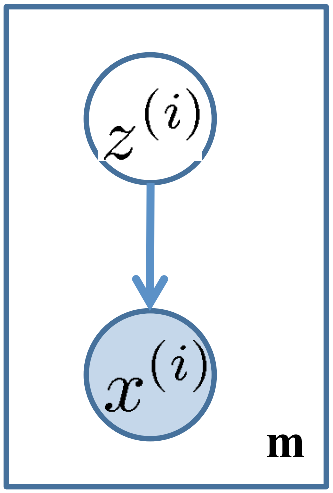
Expectation-Maximization for GMMs

- E-step: compute the expected values of z's
  - Weights $w_{jk} = p(z^{(j)} = k | x^{(j)}, \theta)$

    $= p(z^{(j)} = k, x^{(j)} | \theta) / \sum_{k'} p(z^{(j)} = k, x^{(j)} | \theta)$

- M-step: maximize θ given weights (closed form)
  - $\hat{\pi}_k = \frac{1}{m} \sum w_{jk} = \frac{W_k}{m}$
  - $\hat{\mu}_k = \frac{1}{W_k} \sum_j w_{jk} x^{(j)}$
  - $\hat{\sigma}_k^2 = \frac{1}{W_k} \sum_j w_{jk} (x^{(j)} - \hat{\mu}_k)^2$

Three Mixture Models



Probabilistic PCA

$$\prod_i p(z^{(i)}) = \prod \mathcal{N}(0, I)$$

$$\prod_i p(x^{(i)} | z^{(i)}) = \mathcal{N}(Wz + \mu, \sigma^2 I)$$

Gaussian Mixture Models

$$\prod_i p(z^{(i)}) = [\pi_1 \ldots \pi_K]$$

$$\prod_i p(x^{(i)} | z^{(i)}) = \mathcal{N}(\mu_z, \Sigma_z)$$

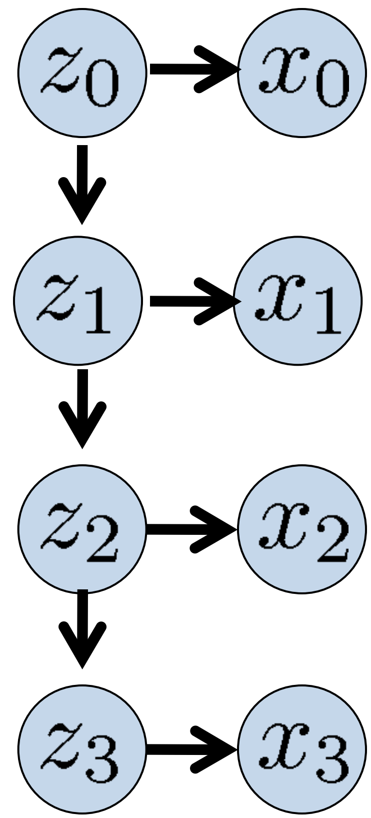
Factor Analysis

$$\prod_i p(z^{(i)}) = \prod \mathcal{N}(0, \Lambda)$$

$$\prod_i p(x^{(i)} | z^{(i)}) = \mathcal{N}(Wz + \mu, \sigma^2 I)$$

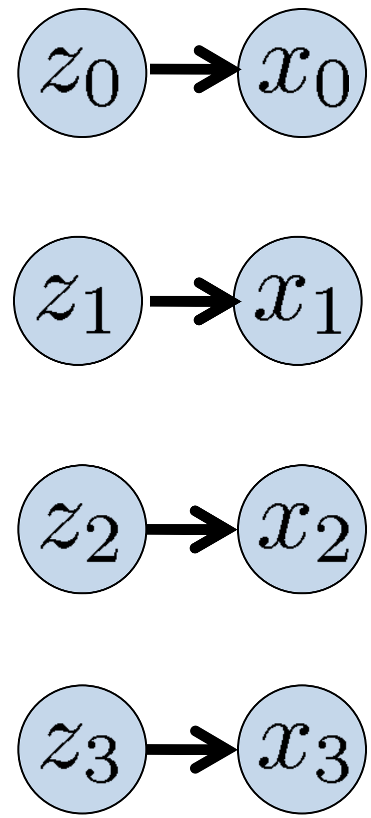
Hidden Markov models

- A Markov model, but in which we can not observe the state (just something indirect)
- Hidden variables Zt
- Observed variables Xt
- “Emission probability distribution”
  - p(Xt = x | Zt = k)



Hidden Markov models

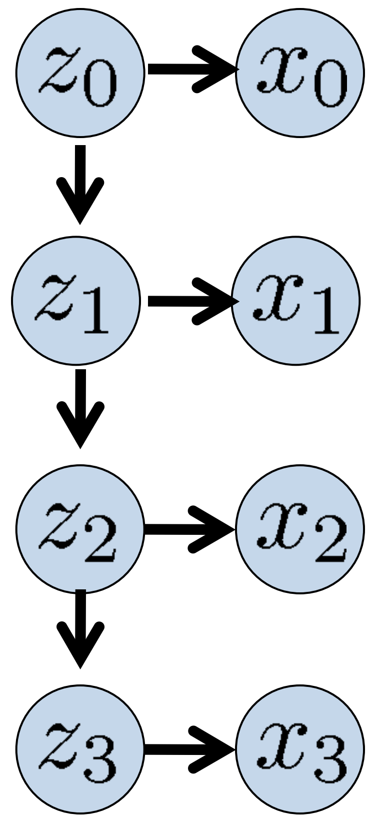
- A Markov model, but in which we can not observe the state (just something indirect)
- Hidden variables Zt
- Observed variables Xt
- "Emission probability distribution"
  - p(Xt = x | Zt = k)



- Special case: Zt independent of Zt-1
  - Discrete distribution over z's
  - Given z, a distribution over obs. x
  - A mixture model, e.g. in clustering

Hidden Markov models

- A Markov model, but in which we can not observe the state (just something indirect)
- Hidden variables Zt
- Observed variables Xt
- "Emission probability distribution"
  - p(Xt = x | Zt = k)



EM algorithm

- E-step: compute

  $q(Z) = p(Z|X ; \theta)$

- M-step: max over θ of

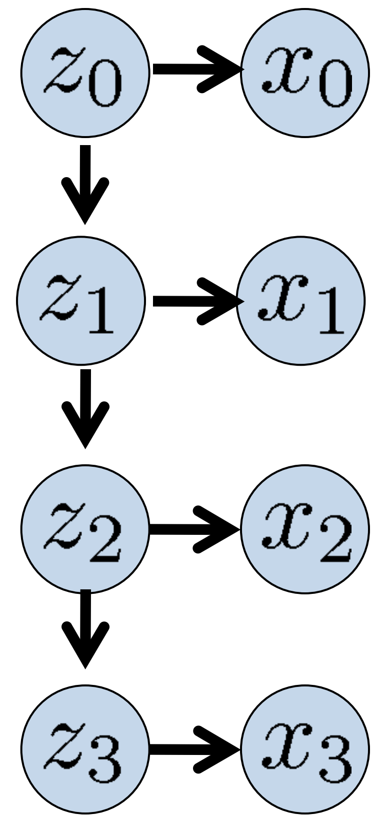
  $E[\log p(Z,X ; \theta)]$

- Sufficient statistics:
  - # times see j->i for each conditional

- Depends on

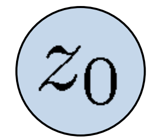
  $q(z_{t+1} = i, z_t = j)$

  $q(z_t = i, x_t = j)$



Inference in HMMs

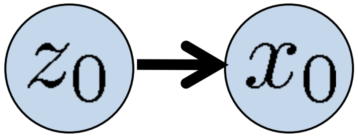
- Need to compute $q(z_t) = p(z_t \mid x_1 \ldots x_n)$
- Two-pass algorithm:
  - Forward pass: $p(z_t = i \mid x_1 \ldots x_t)$
  - Backward pass: $p(x_{t+1} \ldots x_n \mid z_t = i)$



- $p(z_0{=}i) \propto p(z_0{=}i)$

Inference in HMMs

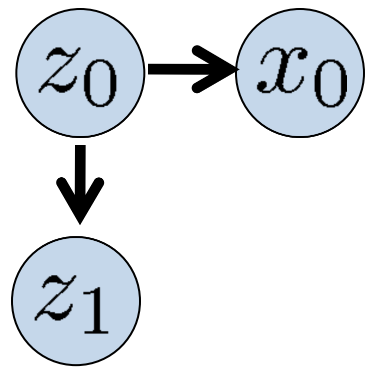
- Need to compute $q(z_t) = p(z_t | x_1 \ldots x_n)$
- Two-pass algorithm:
  - Forward pass: $p(z_t = i \mid x_1 \ldots x_t)$
  - Backward pass: $p(x_{t+1} \ldots x_n \mid z_t = i)$



- $p(z_0=i | x_0) \propto p(z_0=i)\, p(x_0 | z_0=i)$

Inference in HMMs

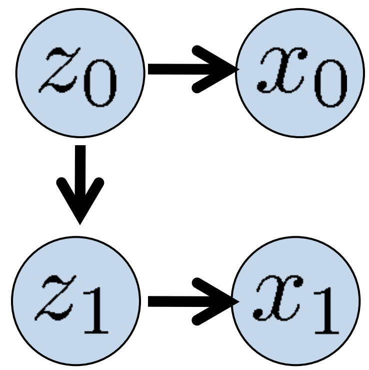
- Need to compute $q(z_t) = p(z_t | x_1 \ldots x_n)$
- Two-pass algorithm:
  - Forward pass: $p(z_t = i \mid x_1 \ldots x_t)$
  - Backward pass: $p(x_{t+1} \ldots x_n \mid z_t = i)$



- $p(z_0=i | x_0) \propto p(z_0=i)\, p(x_0 | z_0=i)$
- $p(z_1=i | x_0) \propto \sum_j p(z_1=i | z_0=j)\, p(z_0=j | x_0)$

Inference in HMMs

- Need to compute $q(z_t) = p(z_t | x_1 \ldots x_n)$
- Two-pass algorithm:
  - Forward pass: $p(z_t = i \mid x_1 \ldots x_t)$
  - Backward pass: $p(x_{t+1} \ldots x_n \mid z_t = i)$

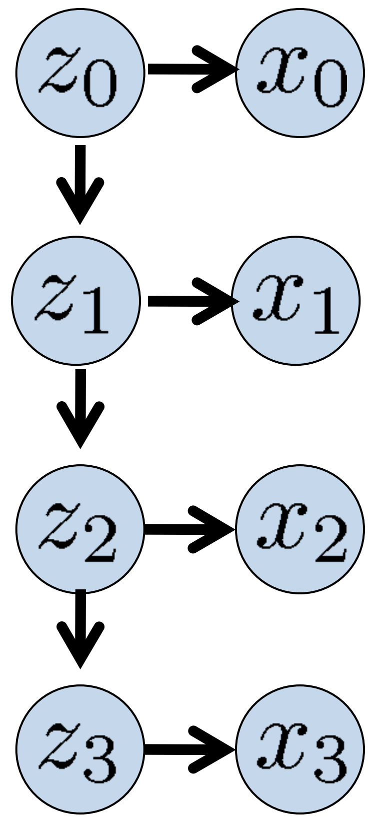


- $p(z_0=i | x_0) \propto p(z_0=i)\, p(x_0 | z_0=i)$
- $p(z_1=i | x_0) \propto \sum_j p(z_1=i | z_0=j)\, p(z_0=j | x_0)$
- $p(z_1=i | x_0, x_1) \propto p(z_1=i | x_0)\, p(x_1 | z_1=i)$

Inference in HMMs

- Need to compute $q(z_t) = p(z_t | x_1 \ldots x_n)$
- Two-pass algorithm:
  - Forward pass: $p(z_t = i \mid x_1 \ldots x_t)$
  - Backward pass: $p(x_{t+1} \ldots x_n \mid z_t = i)$

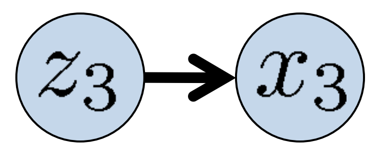
- $p(z_0{=}i | x_0) \propto p(z_0{=}i)\, p(x_0 | z_0{=}i)$
- $p(z_1{=}i | x_0) \propto \sum_j p(z_1{=}i | z_0{=}j)\, p(z_0{=}j | x_0)$
- $p(z_1{=}i | x_0, x_1) \propto p(z_1{=}i | x_0)\, p(x_1 | z_1{=}i)$
- …
- $p(z_n | x_0, \ldots, x_n) \propto p(z_n | x_{0\ldots} x_{n-1})\, p(x_n | z_n)$



Inference in HMMs

- Need to compute $q(z_t) = p(z_t | x_1 \ldots x_n)$
- Two-pass algorithm:
  - Forward pass: $p(z_t = i \mid x_1 \ldots x_t)$
  - Backward pass: $p(x_{t+1} \ldots x_n \mid z_t = i)$

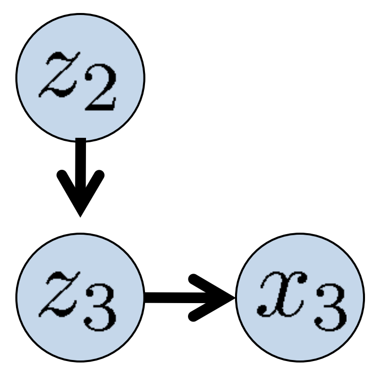
- $p(x_n \mid z_n{=}i)$



Inference in HMMs

- Need to compute $q(z_t) = p(z_t | x_1 \ldots x_n)$
- Two-pass algorithm:
  - Forward pass: $p(z_t = i \mid x_1 \ldots x_t)$
  - Backward pass: $p(x_{t+1} \ldots x_n \mid z_t = i)$

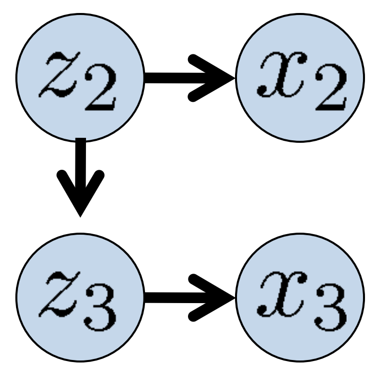
- $p(x_n \mid z_n{=}i)$
- $p(x_n | z_{n-1}{=}i) \propto \sum_j p(z_n{=}j | z_{n-1}{=}i)\, p(x_n | z_n{=}j)$



Inference in HMMs

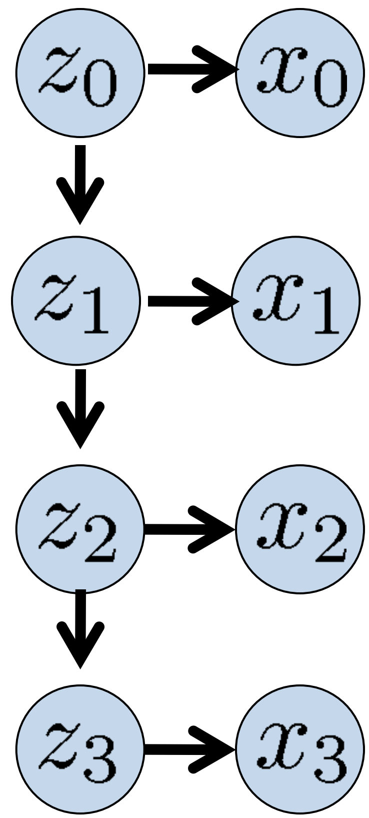
- Need to compute $q(z_t) = p(z_t | x_1 \ldots x_n)$
- Two-pass algorithm:
  - Forward pass: $p(z_t = i \mid x_1 \ldots x_t)$
  - Backward pass: $p(x_{t+1} \ldots x_n \mid z_t = i)$

- $p(x_n \mid z_n = i)$
- $p(x_n | z_{n-1} = i) \propto \sum_j p(z_n = j | z_{n-1} = i)\, p(x_n | z_n = j)$
- $p(x_n, x_{n-1} | z_{n-1} = i) \propto p(x_{n-1} | z_{n-1} = i)\, p(x_n | z_{n-1} = i)$



Inference in HMMs

- Need to compute $q(z_t) = p(z_t | x_1 \ldots x_n)$
- Two-pass algorithm:
  - Forward pass: $p(z_t = i \mid x_1 \ldots x_t)$
  - Backward pass: $p(x_{t+1} \ldots x_n \mid z_t = i)$



- $p(x_n \mid z_n=i)$
- $p(x_n|z_{n-1}=i) \propto \sum_j p(z_n=j|z_{n-1}=i)\, p(x_n|z_n=j)$
- $p(x_n,x_{n-1}|z_{n-1}=i) \propto p(x_{n-1}|z_{n-1}=i)p(x_n|z_{n-1}=i)$
- …

Forward-backward

- $O_t(i) = p(x_t = X_t \mid z_t = i)$
- $T(i,j) = p(z_t = i \mid z_{t-1}=j)$
- Recursion
  - $a_t = (T * a_{t-1}) .* O_t \qquad \propto p(z_t = i \mid x_1 \ldots x_t)$
  - $b_t = T' * (b_{t+1} .* O_{t+1}) \qquad \propto p(x_{t+1} \ldots x_N \mid z_t = i)$
  - $p(z_t = i) \quad \propto a_t(i)\, b_t(i) \quad \propto p(z_t = i \mid x_1 \ldots x_N)$
  - $p(z_t = i, z_{t-1} = j \mid x_1 \ldots x_N) \propto T(i,j)\, b_t(j)\, O_t(j)\, a_{t-1}(i)$

- Gaussian case: similar recursion, "Kalman filter / smoother"

EM for HMMs

- E-step: compute $q(z_t, z_{t-1}) = p(z_t, z_{t-1} | x_1 \ldots x_N)$
  - Using forward-backward recursions
- M-step: maximize expected LL given q(.)

$$\hat{T}_{ij} = \frac{\sum_t q(z_t = i, z_{t-1} = j)}{\sum_{t,k} q(z_t = k, z_{t-1} = j)}$$

(Nij / Nj)

$$\hat{\pi}_i = q(z_0 = i)$$

$$\hat{O}_{iX} = \hat{p}(x_t = X | z_t = i) = \frac{\sum_{t:x_t = X} q(z_t = i)}{\sum_t q(z_t = i)}$$

(Nix / Ni)

EM Variants

- To use EM, we need:
  - Complete log-likelihood easy to optimize
  - $q(z|x) = p(z \mid x ; \theta)$ efficiently computable
- Alternatives? What if $p(z|x)$ is difficult?
- Hard EM: assign "most likely" z
  - If maximizing $p(z \mid x ; \theta)$ is hard, we can approximate
- Stochastic EM: sample $z \sim p(z \mid x ; \theta)$
  - If sampling hard, approximate (e.g., using MCMC)
- Variational EM: approximate $q(z|x)$ directly
  - Replace q(.) with a model that is easier to compute over