# Introduction to Graphical Models

## lecture 10 - Learning with latent variables (EM)

Marc Toussaint
TU Berlin

# Learning Bach

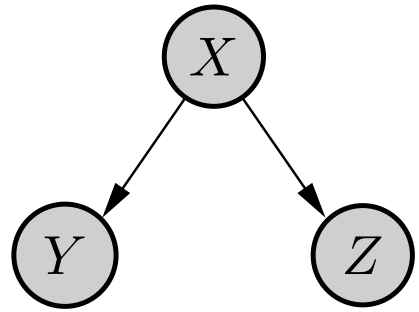
- An example:
  – a machine “listens” (reads notes of) Bach pieces over and over again
  – it’s supposed to learn how to write Bach pieces itself (or at least harmonize them)

- *Harmonizing Chorales in the Style of J S Bach*
  Moray Allan & Chris Williams (NIPS 2004)
  http://www.srcf.ucam.org/~mma29/2002/harmony/

- They learn this by assuming that there is some latent context (the current harmony) implicit in the music

# recap: Learning in graphical models I

- Given three random varibles $X$, $Y$, $Z$ with structure




$$P(X,Y,Z) = P(X)\ P(Y|X)\ P(Z|X)$$

  – unknown parameters $P(X=x) \equiv \pi_x$, $P(Y=y|X=x) \equiv a_{yx}$, $P(Z=z|X=x) \equiv b_{zx}$

  – Given a data set $\{(x_i, y_i, z_i)\}_{i=1}^N$

  – how can we learn the parameters $\theta = (\boldsymbol{\pi}, \boldsymbol{a}, \boldsymbol{b})$?

- answer:

  define counts:

  $c_x = \sum_{i=1}^N [x_i = x]$

  $c_{yx} = \sum_{i=1}^N [y_i = y][x_i = x]$

  $c_{zx} = \sum_{i=1}^N [z_i = z][x_i = x]$

  set parameters equal:

  $\pi_x \leftarrow \frac{c_x}{N}$

  $a_{yx} \leftarrow \frac{c_{yx}}{N c_x}$

  $b_{zx} \leftarrow \frac{c_{zx}}{N c_x}$

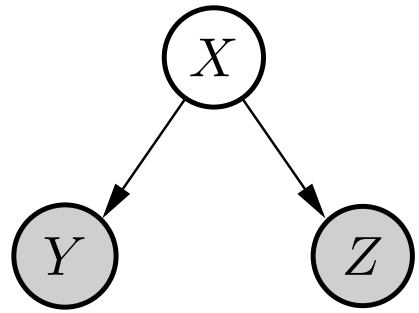
# recap: Learning in graphical models II

- why (in what sense) is this the correct answer?
  – these parameters maximize *observed data log-likelihood*

$$L(\theta) = \log \prod_{i=1}^{n} P(x_i, y_i, z_i \ ; \theta)$$
$$= \sum_{i=1}^{n} \log P(x_i, y_i, z_i \ ; \theta)$$

# Learning with missing data

- Given three random varibles $X$, $Y$, $Z$ with structure




$$P(X, Y, Z) = P(X)\ P(Y|X)\ P(Z|X)$$

  – unknown parameters $P(X=x) \equiv \pi_x$, $P(Y=y|X=x) \equiv a_{yx}$, $P(Z=z|X=x) \equiv b_{zx}$
  – Given a partial data set $\{(y_i, z_i)\}_{i=1}^N$ (observations $x_i$ are missing!)
  – how can we learn the parameters $\theta$?

- any ideas?

# General ideas for learning with missing data

- We should somehow *fill in the missing data*..

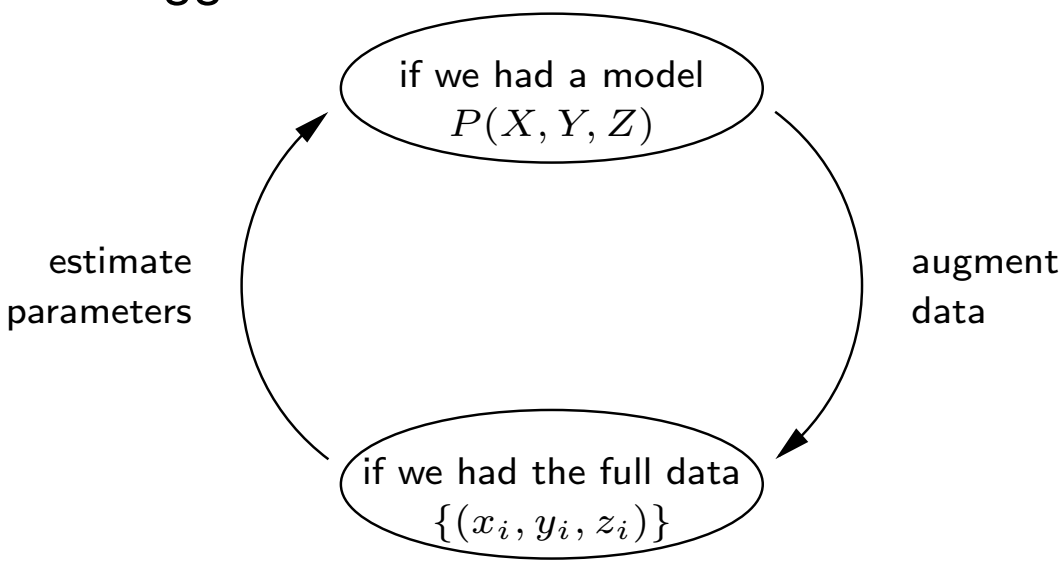
  partial data $\{(y_i, z_i)\}_{i=1}^N$ $\rightarrow$ augmented data $\{(\hat{x}_i, y_i, z_i)\}_{i=1}^N$

- but how should we choose $\hat{x}_i$
  ... invent fictional $\hat{x}_i$ ??

# A chicken and egg problem

- If we knew the model $P(X, Y, Z)$ already, we could use it to invent/estimate an $\hat{x}_i$ for each partial datum $(y_i, z_i)$
  – then use this "augmented data" to train the model

- the chicken and egg situation:



# Viterbi training I

(name Viterbi usually *only* used in context of HMMs, we use it here more generally)

- given partial data $\{(y_i, z_i)\}_{i=1}^N$
- given *some initial* parameters $\theta^{\text{old}}$ that define a model $P(X, Y, Z\ ; \theta^{\text{old}})$
- iterate:
  - for each datum compute $\hat{x}_i = \operatorname{argmax}_x P(x, y_i, z_i\ ; \theta^{\text{old}})$
    ("data agumentation" using the old parameters)
  - using the augmented data $\{(\hat{x}_i, y_i, z_i)\}_{i=1}^N$, compute new parameters

$$\theta^{\text{new}} = \operatorname*{argmax}_\theta L(\theta)$$

that maximize the (augmented) data log-likelihood

# Viterbi training II

- feel uneasy about this algorithm?
  – what if the initial parameters $\theta^{\text{old}}$ are really bad?
  $\rightarrow$ very bad augmented data?
  $\rightarrow$ very bad parameter setting in the next step?

- generally:
  – yes, all these iterative-chicken-and-egg-type algorithms are prone to local minima
  – depend very much on initial choice of parameters

- but we can relax the hard augmentation...

# Expectation Maximization I

- instead of the strict augmentation $\hat{x}_i = \text{argmax}_x P(x, y_i, z_i ; \theta^{\text{old}})$
  we can compute the posterior belief over the missing data using the current model

$$q_i(x) = P(x|y_i, z_i ; \theta^{\text{old}})$$

- but then, how choose new parameters when we have
  – partial data $\{(y_i, z_i)\}_{i=1}^N$ and the posteriors $q_i(x)$ for each $i$??

  $\rightarrow$ we can't use complete data log-likelihood (since data is missing)
  $\rightarrow$ but we can use *expected* data log-likelihood (expectation w.r.t. $q$)!
  $\rightarrow$ maximize expected data log-likelihood (=Expectation Maximization)

- switch notation, to derive general algorithms:

  – let $X$ be a (set of) hidden variables
  – let $Y$ be a (set of) observed variables
  – let $P(X, Y\ ; \theta)$ be a parameterized probabilistic model

# Expected data log-likelihood

- complete data log-likelihood (if $X$ and $Y$ are observed):

$$L(\theta) = \sum_{i=1}^{n} \log P(x_i, y_i \,; \theta)$$

- observed data log-likelihood (if only $Y$ is observed and we can eliminate $X$ analytically):

$$\hat{L}(\theta) = \sum_{i=1}^{n} \log P(y_i \,; \theta) = \sum_{i=1}^{n} \log \sum_{x} P(x, y_i \,; \theta)$$

- expected data log-likelihood (if only $Y$ is observed and we have a posterior $q_i(x; \theta^{\text{old}})$ over the missing data):

$$Q(\theta, \theta^{\text{old}}) = \sum_{i=1}^{n} \sum_{x} q_i(x; \theta^{\text{old}}) \; \log P(x, y_i \,; \theta)$$

$$\text{where} \;\; q_i(x; \theta^{\text{old}}) = P(x | y_i \,; \theta^{\text{old}})$$

**Note:** $Q(\theta, \theta^{\text{old}}) \leq \hat{L}(\theta)$ (see later, free energy)

# Expectation Maximization II

- given partial data $\{y_i\}_{i=1}^N$
- given *some initial* parameters $\theta^{\text{old}}$ that define a model $P(X, Y\ ; \theta^{\text{old}})$
- iterate:

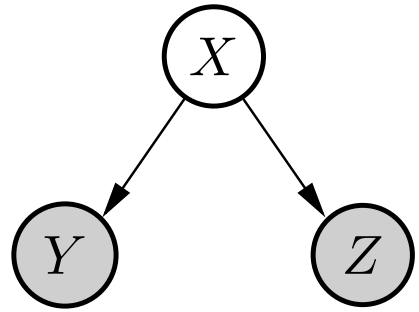
  **(E-step)** for each datum compute $q_i(x; \theta^{\text{old}}) = P(x|y_i\ ; \theta^{\text{old}})$
  ("expected data agumentation" using the old parameters)

  **(M-step)** compute new parameters

  $$\theta^{\text{new}} = \operatorname*{argmax}_{\theta} Q(\theta, \theta^{\text{old}})$$

  that maximize expected data log-likelihood

# Example




$$P(X, Y, Z) = P(X)\ P(Y|X)\ P(Z|X)$$

- **(E-step)** Compute

$$q_i(x\ ; \theta^{\text{old}}) = P(x|y_i, z_i\ ; \theta^{\text{old}})$$
$$\propto P(x)\ P(y_i\,|\,x)\ P(z_i|x) = \pi_x^{\text{old}}\ a_{y_i x}^{\text{old}}\ b_{z_i x}^{\text{old}}$$

  This is an inference problem! – Naive Bayes!

- **(M-step)** compute *expected* counts:

  $c_x = \sum_{i=1}^{N} q_i(x)$
  $c_{yx} = \sum_{i=1}^{N} q_i(x)\ [y_i = y]$
  $c_{zx} = \sum_{i=1}^{N} q_i(x)\ [z_i = z]$

  and set parameter as before

- EM on an intuitive level:
  – fill in missing data
  – do this by computing the posterior $q_i(x\ ; \theta^{\text{old}})$ over missing variables
  – maximized expected data log-likelihood

- now: very elegant and powerful theoretical formulation

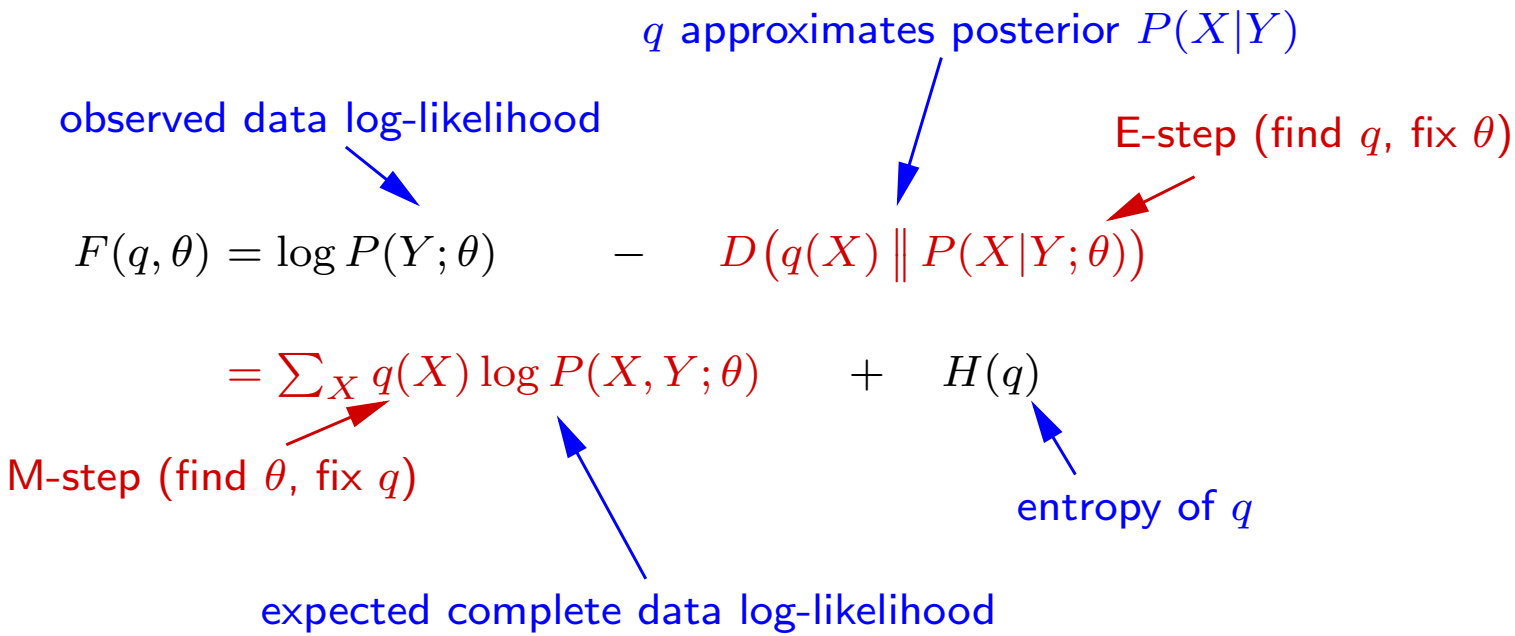
# Free energy view on EM I

- Generally,
  - let $X$ be a (set of i.i.d.) hidden variables
  - let $Y$ be a (set of i.i.d.) observed variables
  - let $P(X, Y\ ; \theta)$ be a parameterized probabilistic model

- define the function

$$
\begin{aligned}
F(q, \theta) &= \log P(Y; \theta) - D\big(q(X) \,\big\|\, P(X|Y; \theta)\big) && (1)\\
&= \log P(Y; \theta) - \sum_X q(X) \log \frac{q(X)}{P(X \mid Y; \theta)} \\
&= \sum_X q(X) \log P(Y; \theta) + \sum_X q(X) \log P(X \mid Y; \theta) + H(q) \\
&= \sum_X q(X) \log P(X, Y; \theta) + H(q), && (2)
\end{aligned}
$$

# Free energy view on EM II



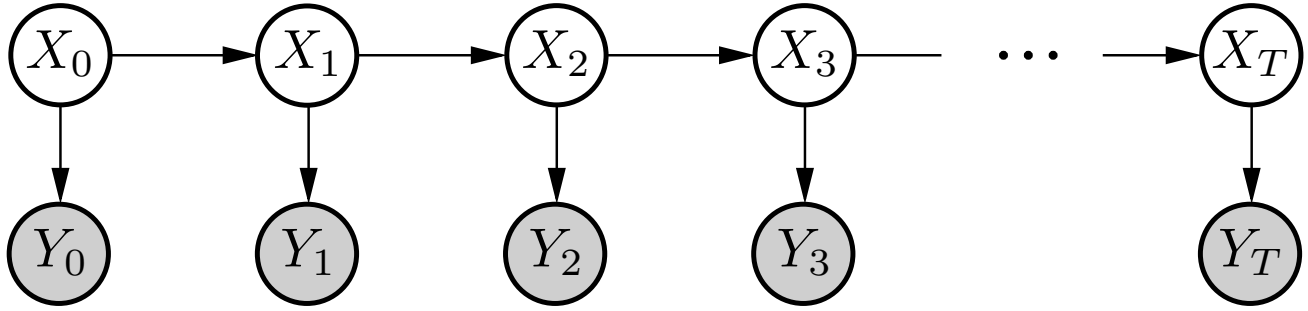

- we actually want to maximize $P(Y;\theta)$ w.r.t. $\theta \rightarrow$ but can't analytically
- instead, maximize lower bound $F(q,\theta) \leq \log P(Y;\theta)$
  - E-step: find $q$ that maximizes $F(q,\theta)$ for fixed $\theta^{\text{old}}$ using (1)
    ($\rightarrow$ find $q$ to minimize KLD, makes lower bound tight for fixed $\theta$)
  - M-step: find $\theta$ that maximizes $F(q,\theta)$ for fixed $q$ using (2)
- EM = step-wise coordinate ascent of the function $F(q,\theta)$

$\Rightarrow$ convergence proof: $F$ can only increase!

# Example

- next time: learning goal-directed behavior
- today: HMMs again

# Example: EM in HMMs




$$P(X_0, .., X_T, Y_0, .., Y_T) = P(X_0) \cdot \prod_{t=1}^{T} P(X_t|X_{t\text{-}1}) \cdot \prod_{t=0}^{T} P(Y_t|X_t)$$

- We've done inference in HMMs enough (e.g., lecture 4)
- Assume we don't know the parameters $\theta = (\boldsymbol{\pi}, \boldsymbol{a}, \boldsymbol{b})$ of the model:

$$P(X_0 = x) = \pi_x$$
$$P(X_t = x'|X_{t\text{-}1} = x) = a_{xx'}$$
$$P(Y_t = y|X_t = x) = b_{xy}$$

- We have a data set of observation sequences $\{y^i_{0:T}\}_{i=1}^N$
  $\rightarrow$ use EM to train parameters (old name of EM for HMMs: *Baum-Welch algorithm*)

# Example: EM in HMMs

- E-step: compute $q_i(x_{0:T}; \theta) = P(x_{0:T} \mid y^i_{0:T}; \theta)$

- M-step: compute

$$\theta^{\text{new}} = \operatorname*{argmax}_{\theta} \sum_{i=1}^{N} \sum_{x_{0:T}} q_i(x_{0:T}; \theta^{\text{old}}) \ \log P(x_{0:T}, y^i_{0:T}; \theta)$$

  Again, this optimization ends up assigning *expected counts* to the new parameters:

$$\pi_x^{\text{new}} = \sum_{i=0}^{N} q_i(x_0 = x)$$

$$a_{xx'}^{\text{new}} = \frac{\sum_{i=0}^{N} \sum_{t=1}^{T} q_i(x_t = x', x_{t\text{-}1} = x)}{\sum_{i=0}^{N} \sum_{t=1}^{T} q_i(x_{t\text{-}1} = x)}$$

$$b_{xy}^{\text{new}} = \frac{\sum_{i=0}^{N} \sum_{t=0}^{T} [y_t^i = y] \ q_i(x_t = x)}{\sum_{i=0}^{N} \sum_{t=0}^{T} q_i(x_t = x)}$$

# back to Bach

(Moray Allan & Chris Williams (NIPS 2004)
http://www.srcf.ucam.org/~mma29/2002/harmony/)

- use an HMM
  – observed sequence $Y_{0:T}$ Soprano melody
  – latent sequence $X_{0:T}$ chord & and harmony:

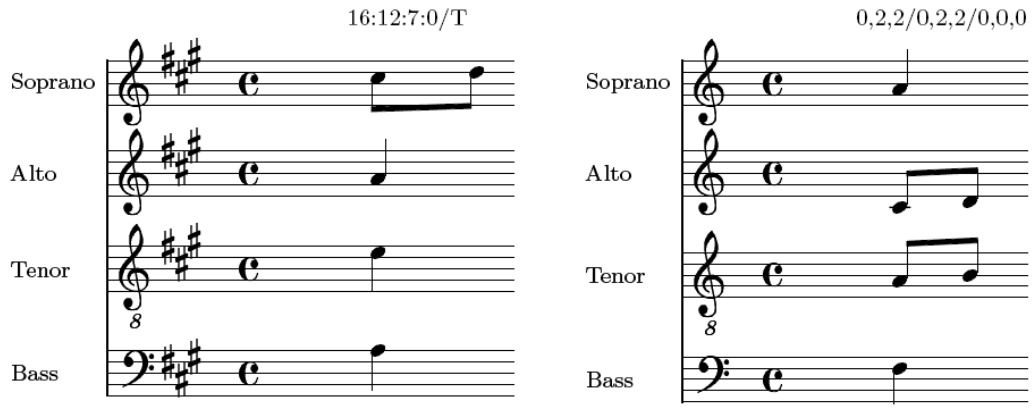



Figure 1: Hidden state representations (a) for harmonisation, (b) for ornamentation.

# back to Bach

- results:

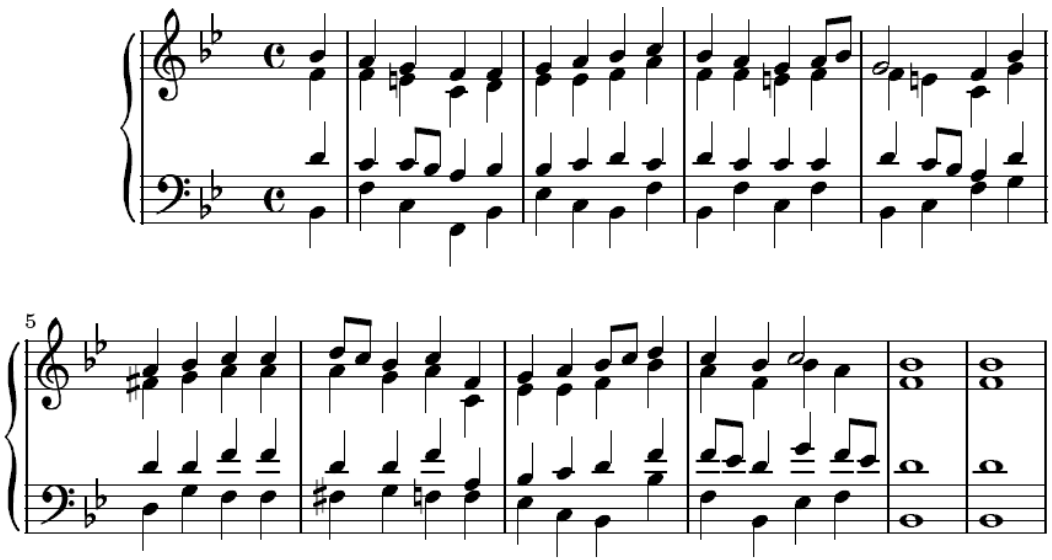



Figure 2: Most likely harmonisation under our model of chorale K4, BWV 48

# summary

- EM $\leftrightarrow$ idea of *fill in missing data*
  EM $\leftrightarrow$ consider *expected* data log-likelihood
  EM $\leftrightarrow$ free energy maximization as lower bound of observed data LL

- 2 core computational principles:
  – information processing
  – learning
  $\rightarrow$ EM combines both for learning with missing data

- next time:
  – learning goal-directed behavior