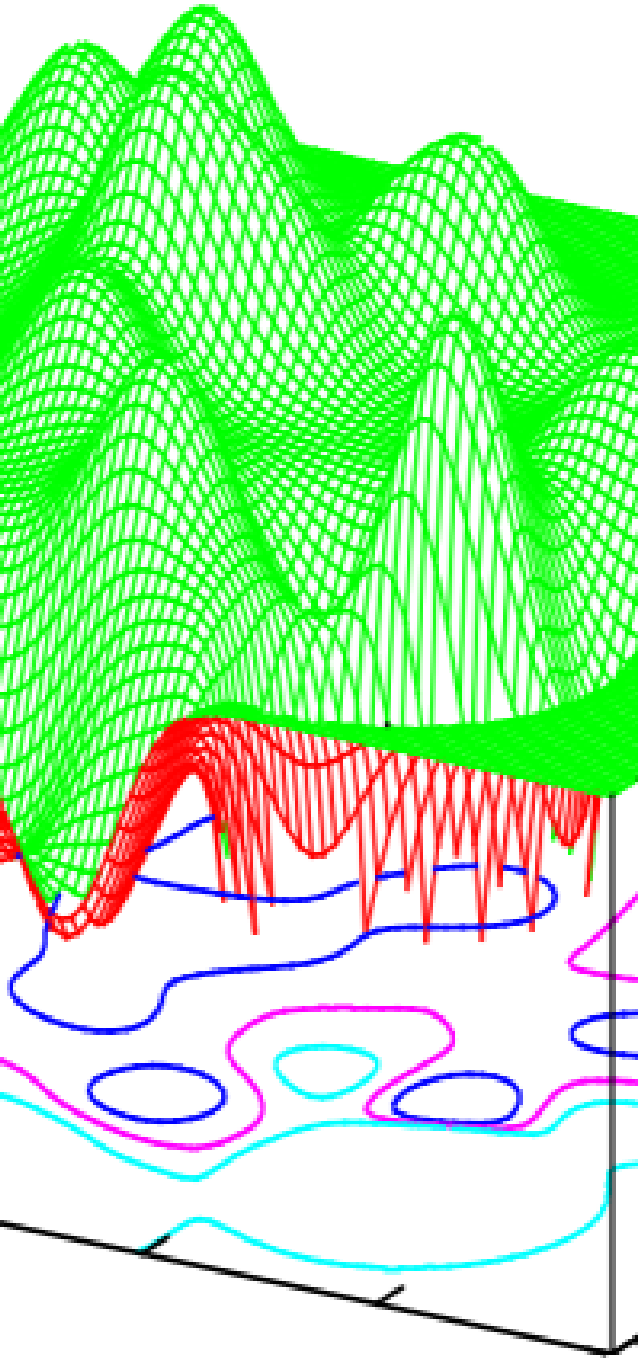


Machine Learning

Learning with Graphical Models

Maximum likelihood learning, Expectation Maximization, Gaussian Mixture Models, Hidden Markov Models, Free Energy formulation, Discriminative Learning

Marc Toussaint
University of Stuttgart
Summer 2014

Fully Bayes vs. ML learning

- **Fully Bayesian learning** (learning = inference)

  Every model "parameter" (e.g., the β of Ridge regression) is a RV. $\rightarrow$ We have a prior over this parameter; learning = computing the posterior

- **Maximum-Likelihood learning** (incl. **Expectation Maximization**)

  The conditional probabilities have parameters θ (e.g., the entries of a CPT). We find parameters θ to *maximize the likelihood of the data under the model*.

Likelihood, ML, fully Bayesian, MAP

We have a probabilistic model $P(X_{1:n}; \theta)$ that depends on parameters θ. (Note: the ';' is used to indicate parameters.) We have data D.

- **The likelihood of the data under the model:**
  $P(D; \theta)$

- **Maximum likelihood (ML) parameter estimate:**
  $$\theta^{\text{ML}} := \operatorname{argmax}_\theta P(D; \theta)$$

- If we treat θ as RV with prior $P(\theta)$ and $P(X_{1:n}, \theta) = P(X_{1:n}|\theta)P(\theta)$:
  Bayesian parameter estimate:
  $$P(\theta|D) \propto P(D|\theta)\ P(\theta)$$

  Bayesian prediction: $P(\text{prediction}|D) = \int_\theta P(\text{prediction}|\theta)\ P(\theta|D)$

- **Maximum a posteriori (MAP) parameter estimate:**
  $$\theta^{\text{MAP}} = \operatorname{argmax}_\theta P(\theta|D)$$

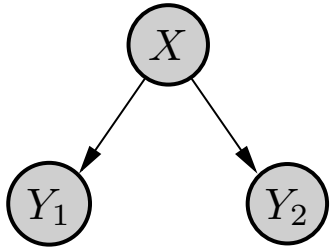
Maximum likelihood learning

– fully observed case

– with latent variables ($\rightarrow$ Expectation Maximization)

Maximum likelihood parameters – fully observed

- An example: Given a model with three random varibles X, Y_1, Y_2:



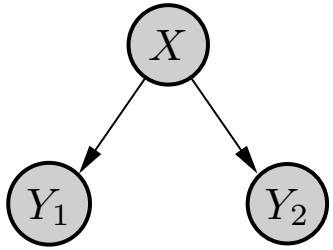
  – We have unknown parameters

$$P(X=x) \equiv a_x,\ P(Y_1=y|X=x) \equiv b_{yx},\ P(Y_2=y|X=x) \equiv c_{yx}$$

  – Given a data set $\{(x_i, y_{1,i}, y_{2,i})\}_{i=1}^n$
  how can we learn the parameters $\theta = (a, b, c)$?

Maximum likelihood parameters – fully observed

- An example: Given a model with three random varibles X, Y_1, Y_2:



  – We have unknown parameters

$$P(X=x) \equiv a_x,\ P(Y_1=y|X=x) \equiv b_{yx},\ P(Y_2=y|X=x) \equiv c_{yx}$$

  – Given a data set $\{(x_i, y_{1,i}, y_{2,i})\}_{i=1}^n$
  how can we learn the parameters $\theta = (a, b, c)$?

- Answer:

$$a_x = \frac{\sum_{i=1}^n [x_i = x]}{n}$$
$$b_{yx} = \frac{\sum_{i=1}^n [y_{1,i} = y \wedge x_i = x]}{na}$$
$$c_{yx} = \frac{\sum_{i=1}^n [y_{2,i} = y \wedge x_i = x]}{na}$$

Maximum likelihood parameters – fully observed

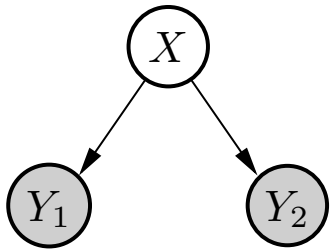
- In what sense is this the correct answer?
  $\rightarrow$ These parameters maximize *observed data log-likelihood*

$$\log P(x_{1:n}, y_{1,1:N}, y_{2,1:n}; \theta) = \sum_{i=1}^{n} \log P(x_i, y_{1,i}, y_{2,i}\ ; \theta)$$

- When all RVs are fully observed, learning ML parameters is usually simple. This also holds when some random variables might be Gaussian (see mixture of Gaussians example).

Maximum likelihood parameters with missing data

- Given a model with three random varibles X, Y_1, Y_2:



  – We have unknown parameters

$$P(X=x) \equiv a_x,\ P(Y_1=y|X=x) \equiv b_{yx},\ P(Y_2=y|X=x) \equiv c_{yx}$$

  – Given a **partial** data set $\{(y_{1,i}, y_{2,i})\}_{i=1}^N$
  how can we learn the parameters $\theta = (a, b, c)$?

General idea

- We should somehow *fill in the missing data*

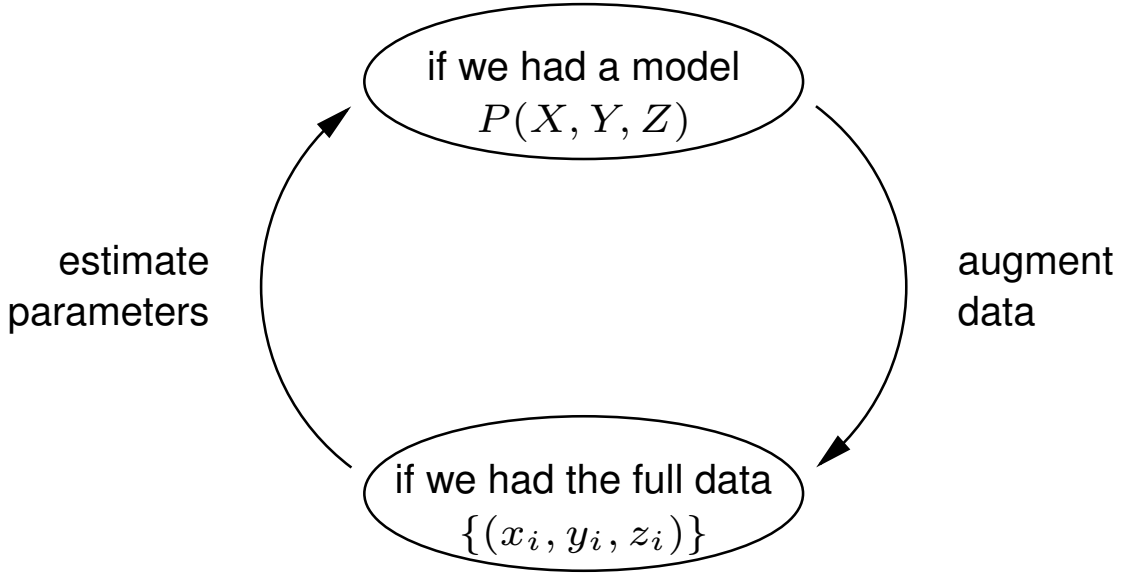
  partial data $\{(y_{1:2,i})\}_{i=1}^n \quad \rightarrow \quad$ augmented data $\{(\hat{x}_i, y_{1:2,i})\}_{i=1}^n$

General idea

- We should somehow *fill in the missing data*
  partial data $\{(y_{1:2,i})\}_{i=1}^n \quad \rightarrow \quad$ augmented data $\{(\hat{x}_i, y_{1:2,i})\}_{i=1}^n$

- If we knew the model $P(X, Y_{1,2})$ already, we could use it to **infer** an $\hat{x}_i$ for each partial datum $(y_{1:2,i})$
  $\rightarrow$ then use this "augmented data" to train the model

- A chicken and egg situation:



Expectation Maximization (EM)

– Let X be a (set of) latent variables
– Let Y be a (set of) observed variables
– Let $P(X, Y; \theta)$ be a parameterized probabilistic model
– We have partial data $D = \{(y_i)\}_{i=1}^n$
– We start with some initial guess θ^{old} of parameters

Iterate:

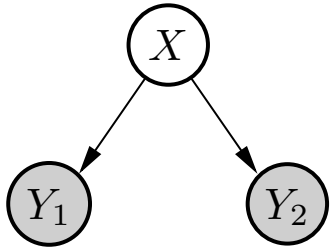
- **Expectation-step:** Compute the posterior belief over the latent variables using θ^{old}

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

- **Maximization-step:** Choose new parameters to maximize the expected data log-likelihood (expectation w.r.t. q)

$$\theta^{\text{new}} = \text{argmax}_\theta \underbrace{\sum_{i=1}^N \sum_x q_i(x) \log P(x, y_i; \theta)}_{=:Q(\theta,q)}$$

EM for our example



- **E-step:** Compute

$$q_i(x\,;\theta^{\text{old}}) = P(x|y_{1:2,i};\theta^{\text{old}}) \propto a_x^{\text{old}}\; b_{y_{1,i}x}^{\text{old}}\; c_{y_{1,i}x}^{\text{old}}$$

- **M-step:** Update using the *expected* counts:

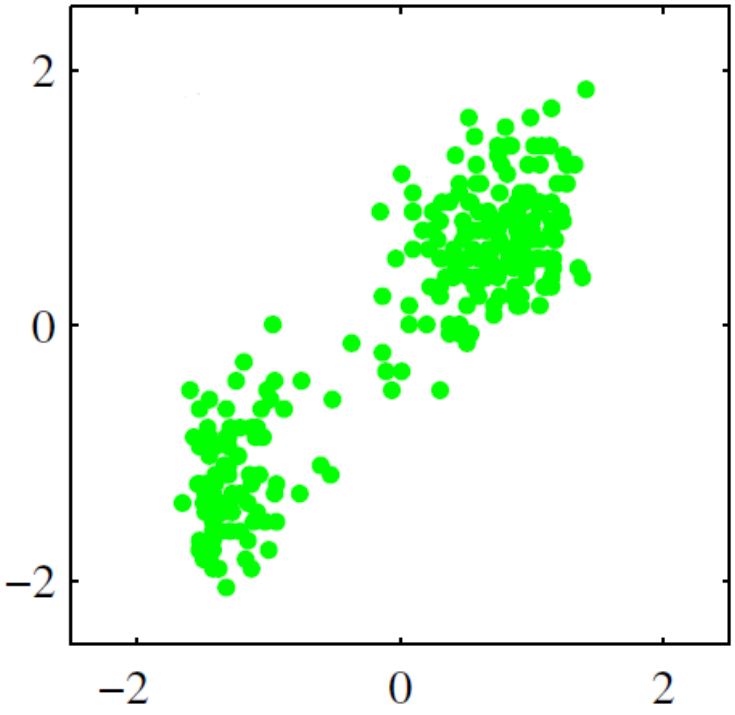
$$a_x^{\text{new}} = \frac{\sum_{i=1}^n q_i(x)}{n}$$
$$b_{yx}^{\text{new}} = \frac{\sum_{i=1}^n q_i(x)\;[y_{1,i}=y]}{na}$$
$$c_{yx}^{\text{new}} = \frac{\sum_{i=1}^n q_i(x)\;[y_{2,i}=y]}{na}$$

Examples for the application of EM

- Mixture of Gaussians
- Hidden Markov Models
- Outlier detection in regression

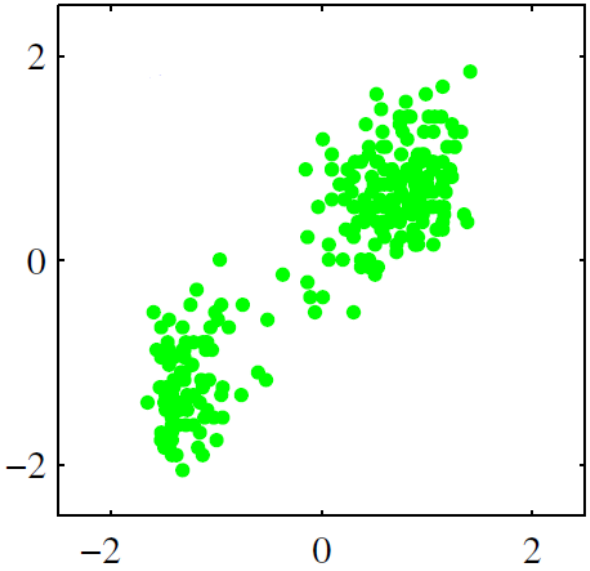
Gaussian Mixture Model – as example for EM

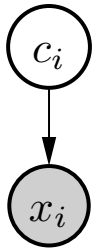
- What might be a *generative* model of this data?



- There might be two “underlying” Gaussian distributions. A data point is drawn (here with about 50/50 chance) by one of the Gaussians.

Gaussian Mixture Model – as example for EM





- K different Gaussians with parmameters μ_k, Σ_k
- A latent random variable $c_i \in \{1, .., K\}$ for each data point with prior
$$P(c_i = k) = \pi_k$$
- The observed data point x_i
$$P(x_i \mid c_i = k; \mu_k, \Sigma_k) = \mathcal{N}(x_i \mid \mu_k, \Sigma_k)$$

Gaussian Mixture Model – as example for EM

Same situation as before:

- If we had full data $D = \{(c_i, x_i)\}_{i=1}^n$, we could estimate parameters

$$\pi_k = \frac{1}{n} \sum_{i=1}^n [c_i = k]$$
$$\mu_k = \frac{1}{n\pi_k} \sum_{i=1}^n [c_i = k]\, x_i$$
$$\Sigma_k = \frac{1}{n\pi_k} \sum_{i=1}^n [c_i = k]\, x_i x_i^\top - \mu_k \mu_k^\top$$

- If we knew the parameters $\mu_{1:K}, \Sigma_{1:K}$, we could infer the mixture labels

$$P(c_i = k \,|\, x_i; \mu_{1:K}, \Sigma_{1:K}) \propto \mathcal{N}(x_i \,|\, \mu_k, \Sigma_k)\; P(c_i = k)$$

Gaussian Mixture Model – as example for EM

- **E-step:** Compute

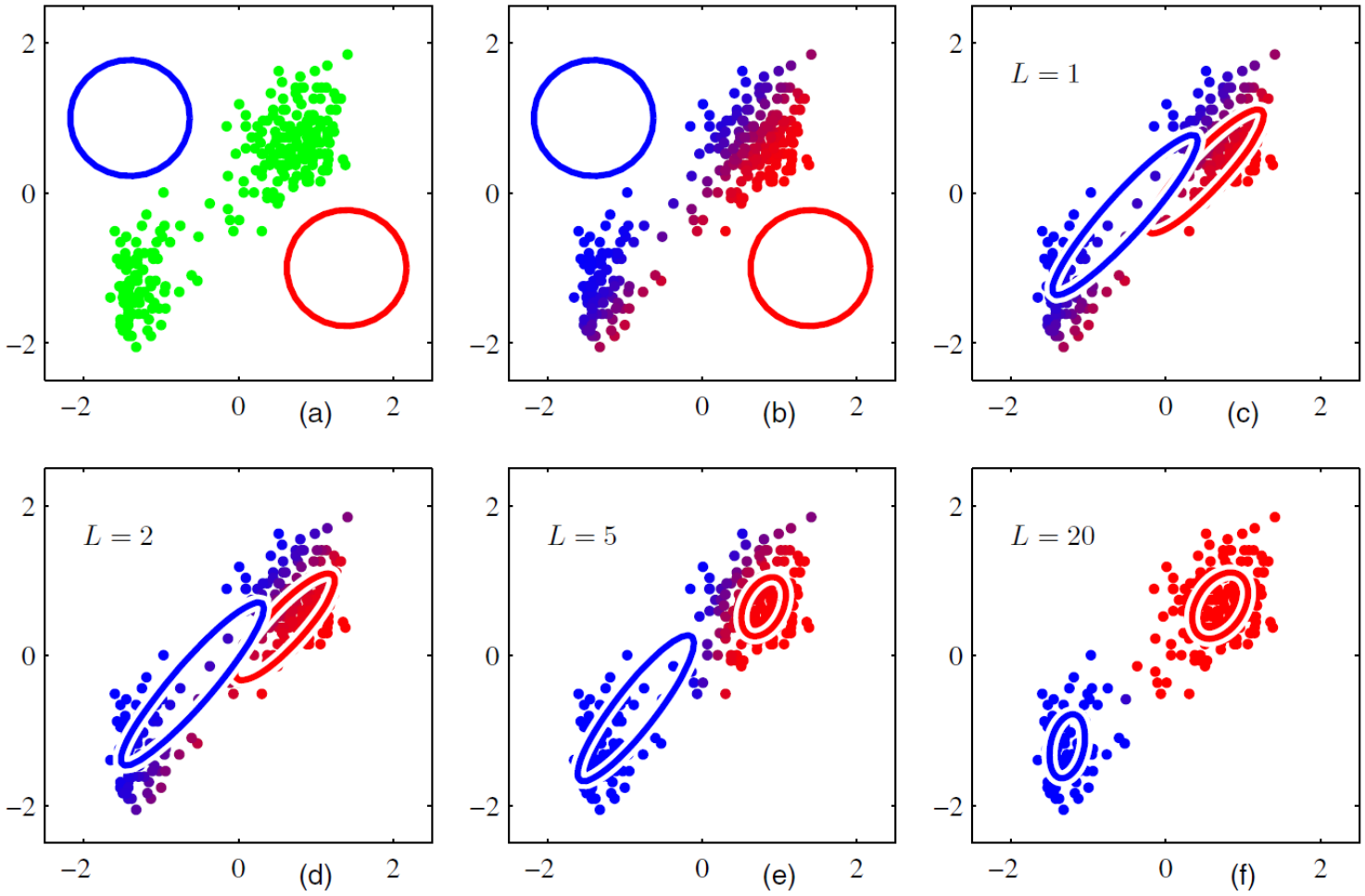
$$q(c_i = k) = P(c_i = k \mid x_i; , \mu_{1:K}, \Sigma_{1:K}) \propto \mathcal{N}(x_i \mid \mu_k, \Sigma_k)\ P(c_i = k)$$

- **M-step:** Update parameters

$$\pi_k = \frac{1}{n} \sum_i q(c_i = k)$$
$$\mu_k = \frac{1}{n\pi_k}\ \sum_i q(c_i = k)\ x_i$$
$$\Sigma_k = \frac{1}{n\pi_k}\ \sum_i q(c_i = k)\ x_i x_i^\top - \mu_k \mu_k^\top$$

Gaussian Mixture Model – as example for EM

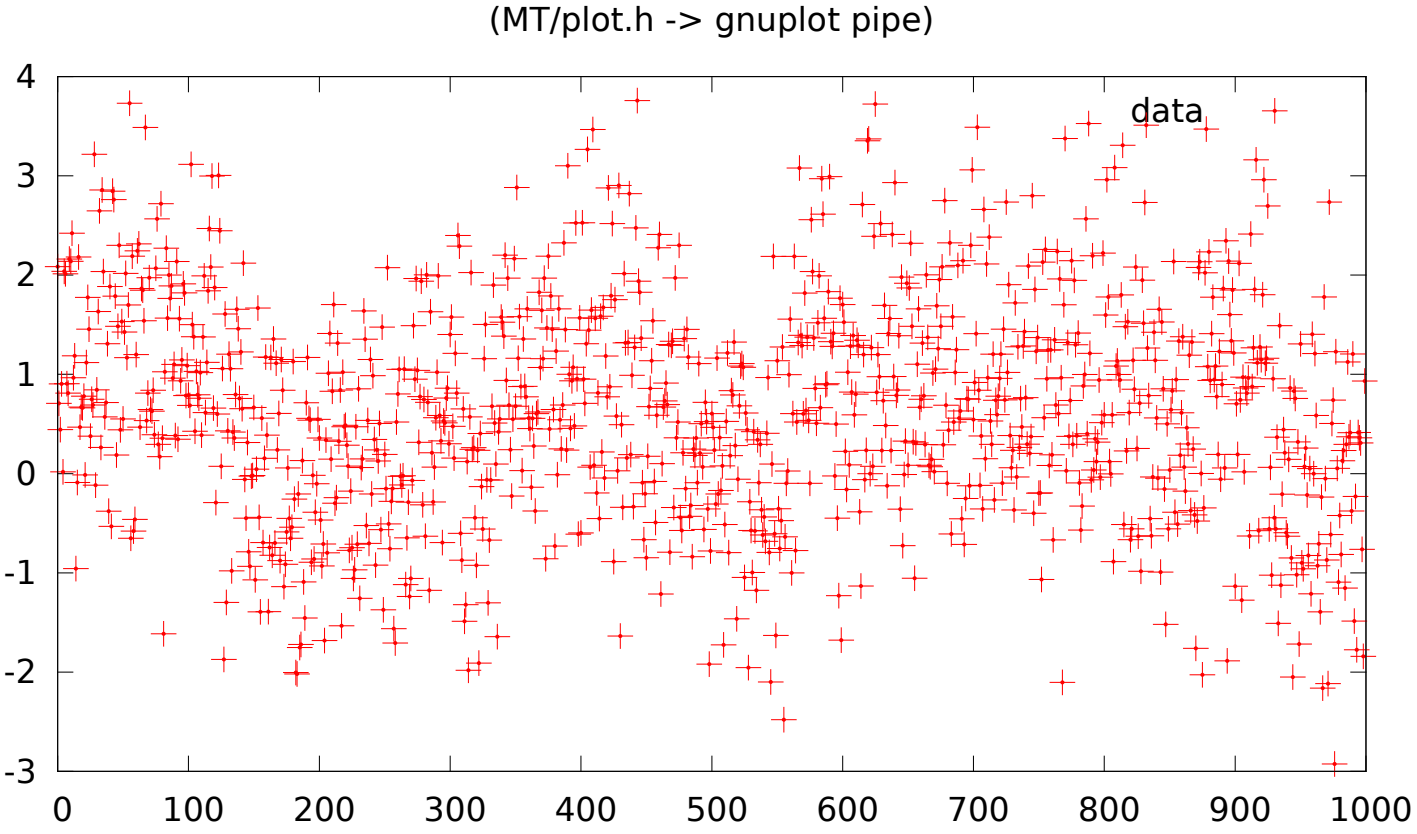
EM iterations for Gaussian Mixture model:



from Bishop

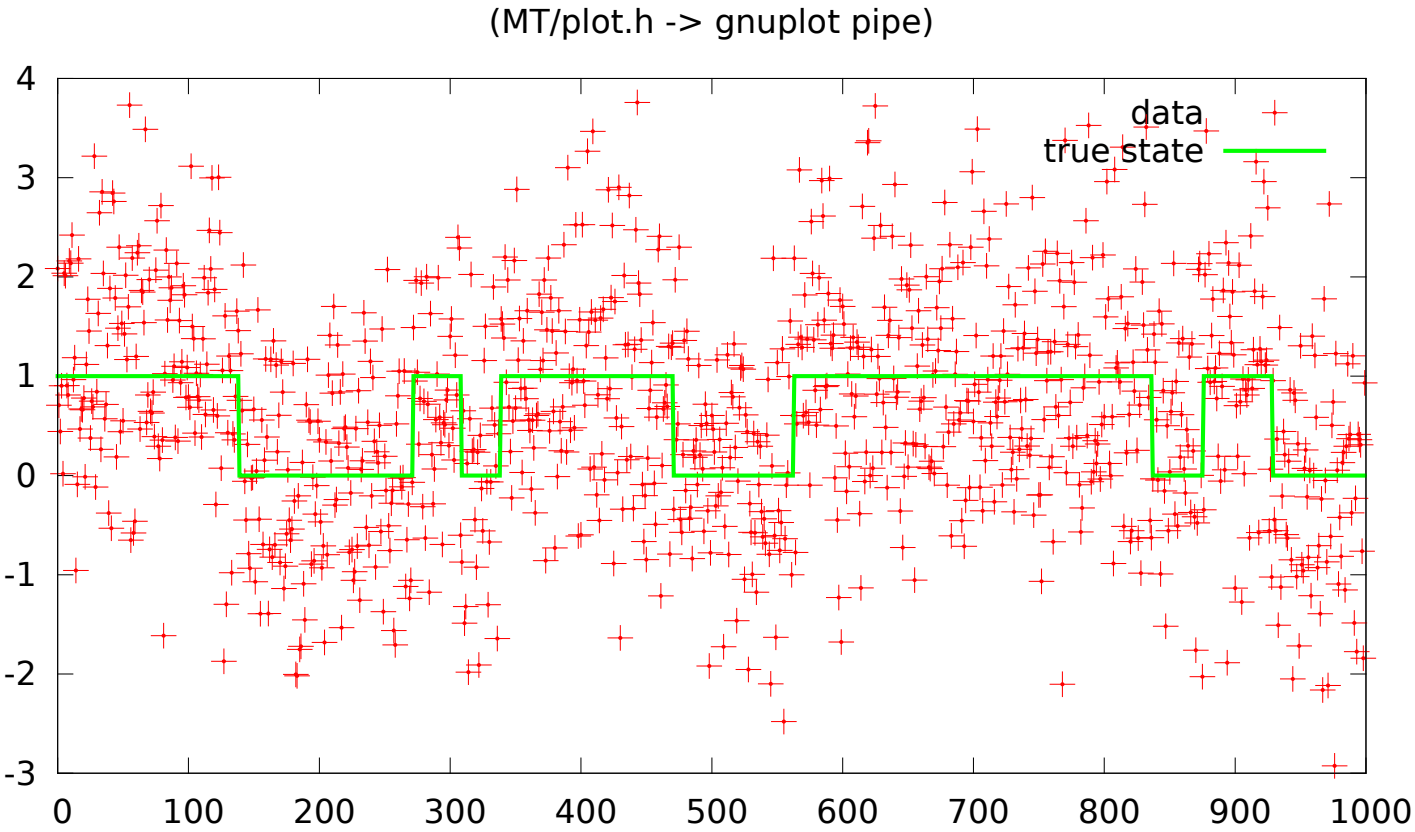
- Generally, the term **mixture** is used when there is a discrete latent variable (which indicates the *“mixture component”*).

Hidden Markov Model – as example for EM



What probabilistic model could explain this data?

Hidden Markov Model – as example for EM



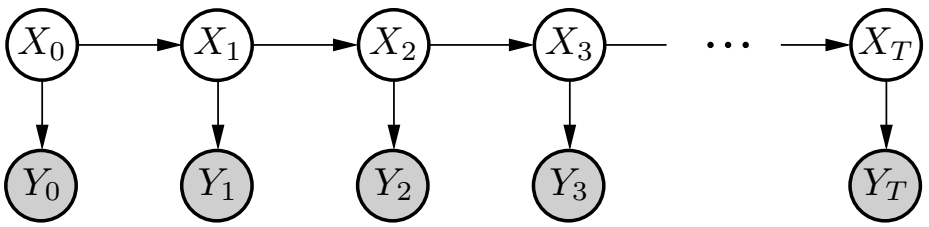
There could be a latent variable, with two states, indicating the high/low mean of the data.

Hidden Markov Models

- We assume we have
  – observed (discrete or continuous) variables Y_t in each time slice
  – a discrete latent variable X_t in each time slice
  – some observation model $P(Y_t \mid X_t; \theta)$
  – some transition model $P(X_t \mid X_{t\text{-}1}; \theta)$

- A **Hidden Markov Model (HMM)** is defined as the joint distribution

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



EM for Hidden Markov Models

- Given data $D = \{y_t\}_{t=0}^{T}$. How can we learn all parameters of a HMM?

EM for Hidden Markov Models

- Given data $D = \{y_t\}_{t=0}^T$. How can we learn all parameters of a HMM?

- What are the parameters of an HMM?
  - The number K of discrete latent states x_t (the cardinality of dom(x_t))
  - The transition probability $P(x_t|x_{t\text{-}1})$ (a $K \times K$-CPT $P_{k'k}$)
  - The initial state distribution $P(x_0)$ (a K-table π_k)
  - Parameters of the output distribution $P(y_t \,|\, x_t = k)$
    e.g., the mean μ_k and covariance matrix Σ_k for each latent state k

EM for Hidden Markov Models

- Given data $D = \{y_t\}_{t=0}^T$. How can we learn all parameters of a HMM?

- What are the parameters of an HMM?
  – The number K of discrete latent states x_t (the cardinality of dom(x_t))
  – The transition probability $P(x_t|x_{t\text{-}1})$ (a $K \times K$-CPT $P_{k'k}$)
  – The initial state distribution $P(x_0)$ (a K-table π_k)
  – Parameters of the output distribution $P(y_t \,|\, x_t = k)$
  e.g., the mean μ_k and covariance matrix Σ_k for each latent state k

- Learning K is very hard!
  – Try and evaluate many K
  – Have a distribution over K ($\rightarrow$ infinite HMMs)
  – We assume K fixed in the following

- Learning the other parameters is easier: **Expectation Maximization**

EM for Hidden Markov Models

The paramers of an HMM are $\theta = (P_{k'k}, \pi_k, \mu_k, \Sigma_k)$

- **E-step:** Compute the marginal posteriors (also pair-wise)

$$q(x_t) = P(x_t \,|\, y_{1:T}; \theta^{\text{old}})$$
$$q(x_t, x_{t+1}) = P(x_t, x_{t+1} \,|\, y_{1:T}; \theta^{\text{old}})$$

- **M-step:** Update the parameters

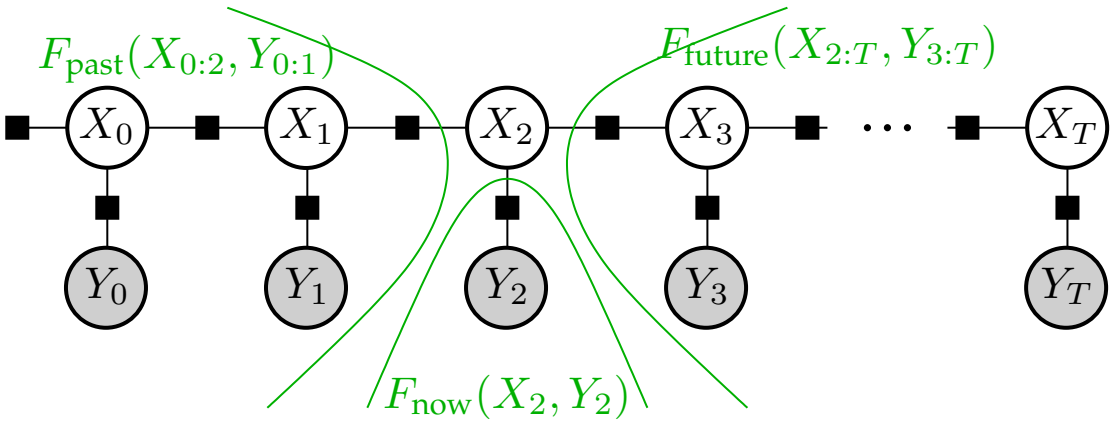
$$P_{k'k} \leftarrow \frac{\sum_{t=0}^{T\text{-}1} q(x_t = k, x_{t+1} = k')}{\sum_{t=0}^{T\text{-}1} q(x_t = k)} \;, \qquad \pi_k \leftarrow q(x_0 = k)$$

$$\mu_k \leftarrow \sum_{t=0}^{T} w_{kt} y_t \;, \qquad \Sigma_k \leftarrow \sum_{t=0}^{T} w_{kt}\; y_t y_t^\top - \mu_k \mu_k^\top$$

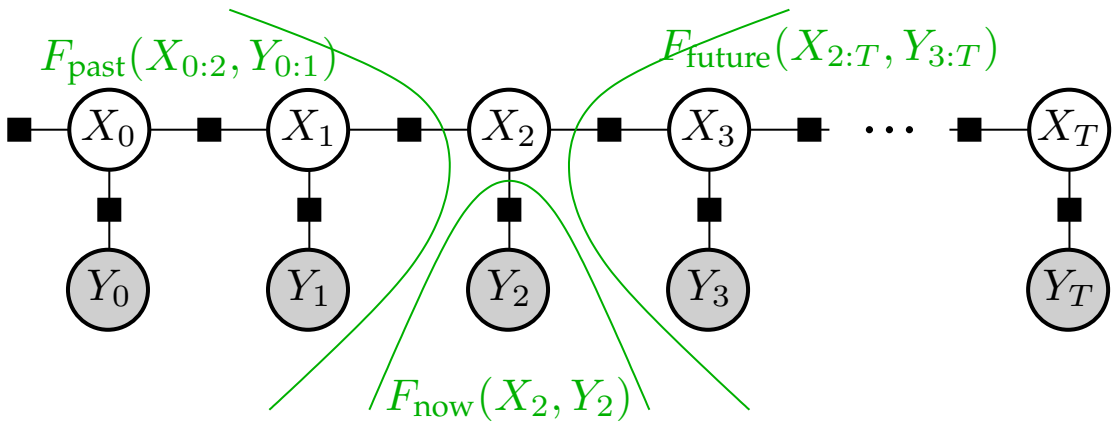
with $w_{kt} = q(x_t = k)/(\sum_{t=0}^{T} q(x_t = k))$ (each row of w_{kt} is normalized)

compare with EM for Gaussian Mixtures!

E-step: Inference in an HMM – a tree!



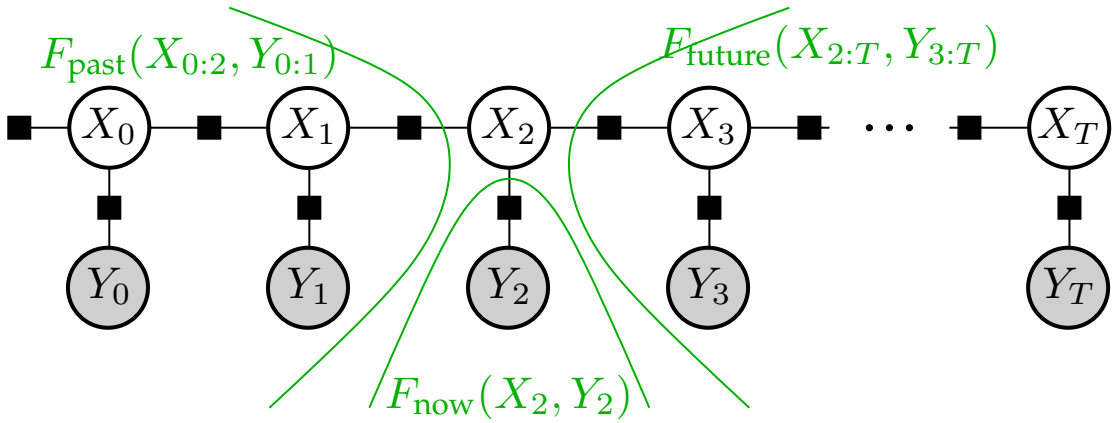
E-step: Inference in an HMM – a tree!



- The marginal posterior $P(X_t \mid Y_{1:T})$ is the product of three messages

$$P(X_t \mid Y_{1:T}) \propto P(X_t, Y_{1:T}) = \underbrace{\mu_{\text{past}}}_{\alpha}(X_t)\ \underbrace{\mu_{\text{now}}}_{\varrho}(X_t)\ \underbrace{\mu_{\text{future}}}_{\beta}(X_t)$$

E-step: Inference in an HMM – a tree!



- The marginal posterior $P(X_t \mid Y_{1:T})$ is the product of three messages

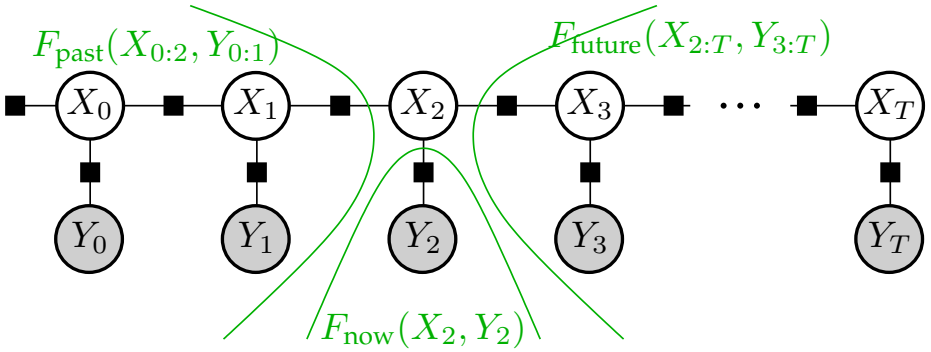
$$P(X_t \mid Y_{1:T}) \propto P(X_t, Y_{1:T}) = \underbrace{\mu_{\text{past}}}_{\alpha}(X_t)\ \underbrace{\mu_{\text{now}}}_{\varrho}(X_t)\ \underbrace{\mu_{\text{future}}}_{\beta}(X_t)$$

- For all $a < t$ and $b > t$
  – X_a conditionally independent from X_b given X_t
  – Y_a conditionally independent from Y_b given X_t

  "The future is independent of the past given the present"

  Markov property

Inference in HMMs



Applying the general message passing equations:

forward msg. $$\mu_{X_{t\text{-}1} \to X_t}(x_t) =: \alpha_t(x_t) = \sum_{x_{t\text{-}1}} P(x_t | x_{t\text{-}1})\ \alpha_{t\text{-}1}(x_{t\text{-}1})\ \varrho_{t\text{-}1}(x_{t\text{-}1})$$

$$\alpha_0(x_0) = P(x_0)$$

backward msg. $$\mu_{X_{t+1} \to X_t}(x_t) =: \beta_t(x_t) = \sum_{x_{t+1}} P(x_{t+1} | x_t)\ \beta_{t+1}(x_{t+1})\ \varrho_{t+1}(x_{t+1})$$

$$\beta_T(x_0) = 1$$

observation msg. $$\mu_{Y_t \to X_t}(x_t) =: \varrho_t(x_t) = P(y_t \mid x_t)$$

posterior marginal $$q(x_t) \propto \alpha_t(x_t)\ \varrho_t(x_t)\ \beta_t(x_t)$$

posterior marginal $$q(x_t, x_{t+1}) \propto \alpha_t(x_t)\ \varrho_t(x_t)\ P(x_{t+1} | x_t)\ \varrho_{t+1}(x_{t+1})\ \beta_{t+1}(x_{t+1})$$

Inference in HMMs – implementation notes

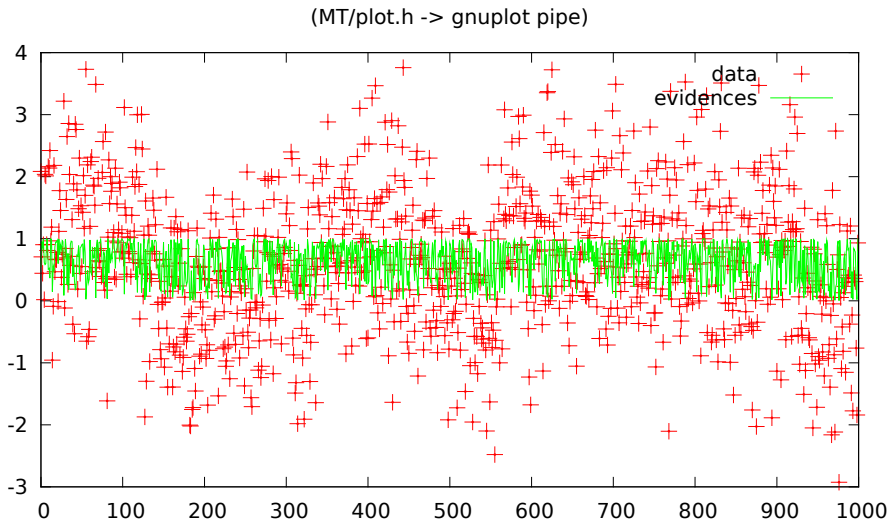
- The message passing equations can be implemented by reinterpreting them as matrix equations: Let $\boldsymbol{\alpha}_t, \boldsymbol{\beta}_t, \boldsymbol{\varrho}_t$ be the vectors corresponding to the probability tables $\alpha_t(x_t), \beta_t(x_t), \varrho_t(x_t)$; and let $\boldsymbol{P}$ be the matrix with enties $P(x_t \mid x_{t\text{-}1})$. Then

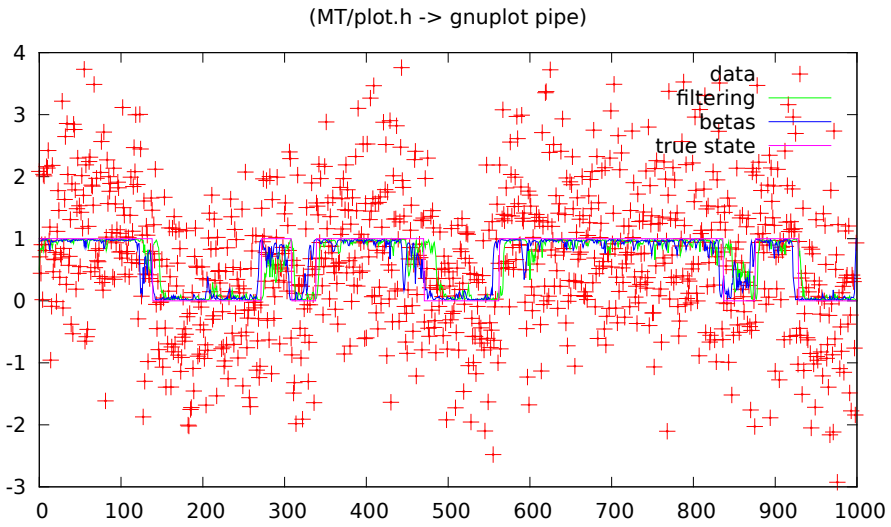
---

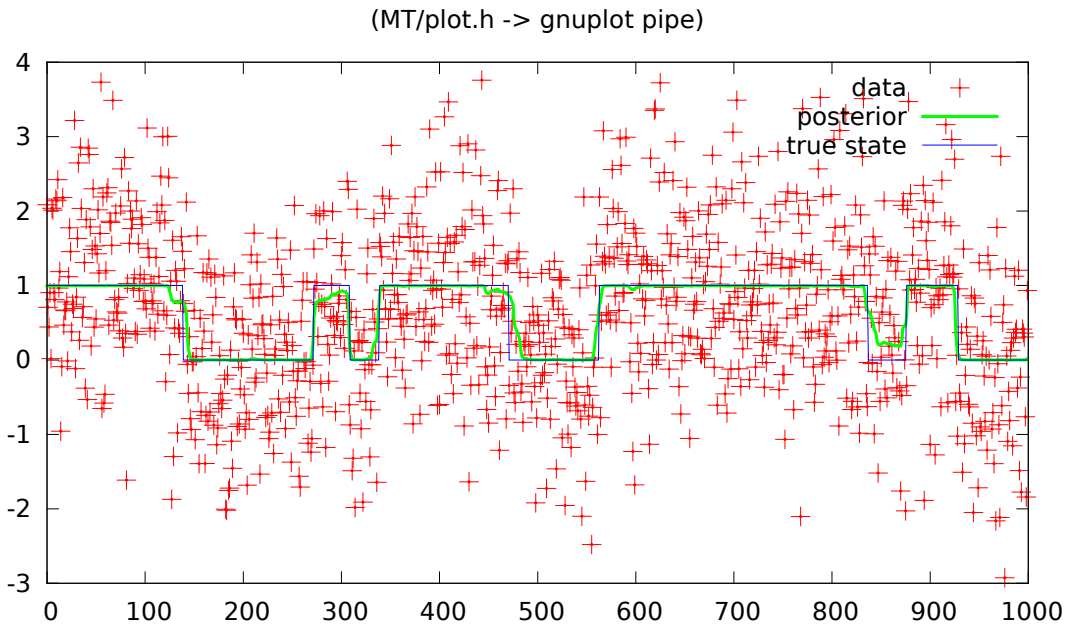
1: $\boldsymbol{\alpha}_0 = \boldsymbol{\pi},\ \boldsymbol{\beta}_T = 1$
2: $\texttt{for}_{t=1:T\text{-}1}:\ \boldsymbol{\alpha}_t = \boldsymbol{P}\,(\boldsymbol{\alpha}_{t\text{-}1} \cdot \boldsymbol{\varrho}_{t\text{-}1})$
3: $\texttt{for}_{t=T\text{-}1:0}:\ \boldsymbol{\beta}_t = \boldsymbol{P}^\top(\boldsymbol{\beta}_{t+1} \cdot \boldsymbol{\varrho}_{t+1})$
4: $\texttt{for}_{t=0:T}:\ \boldsymbol{q}_t = \boldsymbol{\alpha}_t \cdot \boldsymbol{\varrho}_t \cdot \boldsymbol{\beta}_t$
5: $\texttt{for}_{t=0:T\text{-}1}:\ \boldsymbol{Q}_t = \boldsymbol{P} \cdot [(\boldsymbol{\beta}_{t+1} \cdot \boldsymbol{\varrho}_{t+1})\,(\boldsymbol{\alpha}_t \cdot \boldsymbol{\varrho}_t)^\top]$

---

where $\cdot$ is the *element-wise product!* Here, $\boldsymbol{q}_t$ is the vector with entries $q(x_t)$, and $\boldsymbol{Q}_t$ the matrix with entries $q(x_{t+1}, x_t)$. Note that the equation for $\boldsymbol{Q}_t$ describes $Q_t(x', x) = P(x'|x)[(\beta_{t+1}(x')\varrho_{t+1}(x'))(\alpha_t(x)\varrho_t(x))]$.


(MT/plot.h -> gnuplot pipe)
data
evidences
4
3
2
1
0
-1
-2
-3
0
100
200
300
400
500
600
700
800
900
1000


(MT/plot.h -> gnuplot pipe)
data
filtering
betas
true state
4
3
2
1
0
-1
-2
-3
0
100
200
300
400
500
600
700
800
900
1000


(MT/plot.h -> gnuplot pipe)
data
posterior
true state
4
3
2
1
0
-1
-2
-3
0
100
200
300
400
500
600
700
800
900
1000

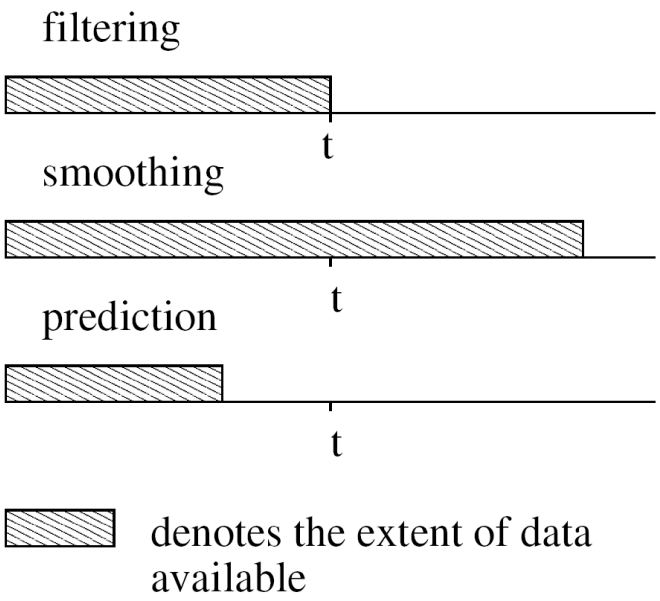
Inference in HMMs: classical derivation

Given our knowledge of Belief propagation, inference in HMMs is simple. For reference, here is a more classical derivation:

$$
\begin{aligned}
P(x_t \,|\, y_{0:T}) &= \frac{P(y_{0:T} \,|\, x_t)\; P(x_t)}{P(y_{0:T})} \\
&= \frac{P(y_{0:t} \,|\, x_t)\; P(y_{t+1:T} \,|\, x_t)\; P(x_t)}{P(y_{0:T})} \\
&= \frac{P(y_{0:t}, x_t)\; P(y_{t+1:T} \,|\, x_t)}{P(y_{0:T})} \\
&= \frac{\alpha_t(x_t)\; \beta_t(x_t)}{P(y_{0:T})} \\
\alpha_t(x_t) &:= P(y_{0:t}, x_t) = P(y_t|x_t)\; P(y_{0:t\text{-}1}, x_t) \\
&= P(y_t|x_t) \sum_{x_{t\text{-}1}} P(x_t \,|\, x_{t\text{-}1})\; \alpha_{t\text{-}1}(x_{t\text{-}1}) \\
\beta_t(x_t) &:= P(y_{t+1:T} \,|\, x_t) = \sum_{x+1} P(y_{t+1:T} \,|\, x_{t+1})\; P(x_{t+1} \,|\, x_t) \\
&= \sum_{x+1} \Big[\beta_{t+1}(x_{t+1})\; P(y_{t+1}|x_{t+1})\Big]\; P(x_{t+1} \,|\, x_t)
\end{aligned}
$$

Note: α_t here is the same as $\alpha_t \cdot \varrho_t$ on all other slides!

Different inference problems in HMMs



- $P(x_t \mid y_{0:T})$ marginal posterior
- $P(x_t \mid y_{0:t})$ filtering
- $P(x_t \mid y_{0:a})$, $t > a$ prediction
- $P(x_t \mid y_{0:b})$, $t < b$ smoothing
- $P(y_{0:T})$ likelihood calculation

- Viterbi alignment: Find sequence $x^*_{0:T}$ that maximizes $P(x_{0:T} \mid y_{0:T})$
  (This is done using max-product, instead of sum-product message passing.)

HMM remarks

- The computation of forward and backward messages along the Markov chain is also called **forward-backward algorithm**
- The EM algorithm to learn the HMM parameters is also called **Baum-Welch algorithm**
- If the latent variable x_t is **continuous** $x_t \in \mathbb{R}^d$ instead of discrete, then such a Markov model is also called *state space model*.
- If the continuous transitions are linear Gaussian

$$P(x_{t+1}|x_t) = \mathcal{N}(x_{t+1}|Ax_t + a, Q)$$

  then the forward and backward messages α_t and β_t are also Gaussian.
  $\rightarrow$ forward filtering is also called **Kalman filtering**
  $\rightarrow$ smoothing is also called **Kalman smoothing**
- Sometimes, computing forward and backward messages (in disrete or continuous context) is also called **Bayesian filtering/smoothing**

HMM example: Learning Bach

- A machine "listens" (reads notes of) Bach pieces over and over again $\rightarrow$ 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)

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

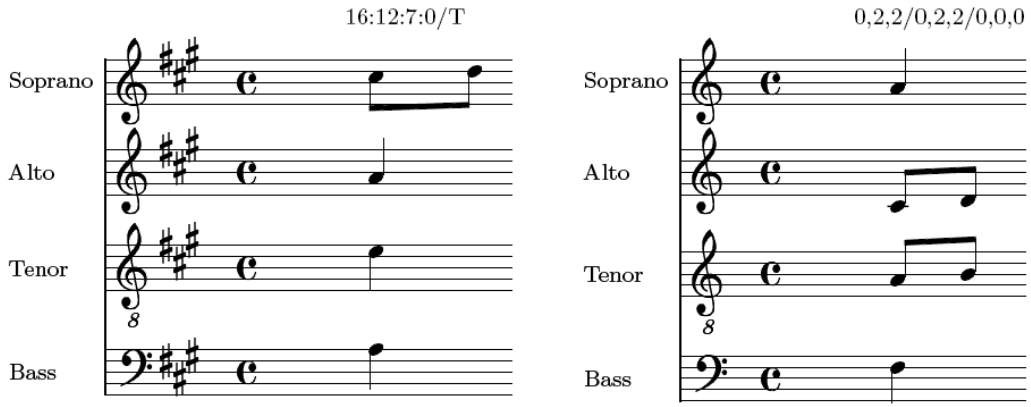


Figure 1: Hidden state representations (a) for harmonisation, (b) for ornamentation.

HMM example: Learning Bach

- results: `http://www.anc.inf.ed.ac.uk/demos/hmmbach/`

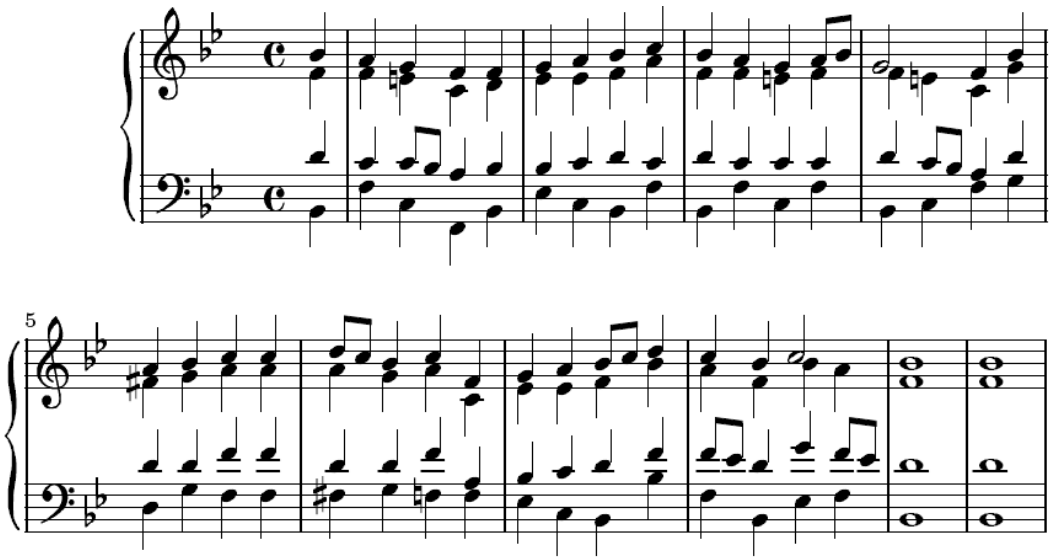


Figure 2: Most likely harmonisation under our model of chorale K4, BWV 48

- See also work by Gerhard Widmer
  `http://www.cp.jku.at/people/widmer/`

Free Energy formulation of EM

- We introduced EM rather heuristically: an iteration to resolve the chicken and egg problem.
  Are there convergence proofs?
  Are there generalizations?
  Are there more theoretical insights?

  $\rightarrow$ Free Energy formulation of Expectation Maximization

Free Energy formulation of EM

- Define a function (called "Free Energy")

$$F(q,\theta) = \log P(Y;\theta) - D\big(q(X) \,\big\|\, P(X|Y;\theta)\big)$$

where

$$\log P(Y;\theta) = \log \sum_X P(X,Y;\theta) \qquad \text{log-likelihood of } \textit{observed} \text{ data } Y$$

$$D\big(q(X) \,\big\|\, P(X|Y;\theta)\big) := \sum_X q(X) \log \frac{q(X)}{P(X \mid Y;\theta)} \qquad \text{Kullback-Leibler div.}$$

- The Kullback-Leibler divergence is sort-of a distance measure between two distributions: $D\big(q \,\big\|\, p\big)$ is always positive and zero only if $q = p$.

Free Energy formulation of EM

- We can write the free energy in two ways

$$F(q,\theta) = \underbrace{\log P(Y;\theta)}_{\text{data log-like.}} - \underbrace{D\big(q(X) \,\|\, P(X|Y;\theta)\big)}_{\text{E-step: } q \leftarrow \operatorname{argmin}} \tag{1}$$

$$= \log P(Y;\theta) - \sum_X q(X) \log \frac{q(X)}{P(X \mid Y;\theta)}$$

$$= \sum_X q(X) \log P(Y;\theta) + \sum_X q(X) \log P(X \mid Y;\theta) + H(q)$$

$$= \underbrace{\sum_X q(X) \log P(X,Y;\theta)}_{\text{M-step: } \theta \leftarrow \operatorname{argmax}} + H(q), \tag{2}$$

Free Energy formulation of EM

- We can write the free energy in two ways

$$F(q,\theta) = \underbrace{\log P(Y;\theta)}_{\text{data log-like.}} - \underbrace{D\big(q(X) \,\|\, P(X|Y;\theta)\big)}_{\text{E-step: } q \leftarrow \operatorname{argmin}} \tag{1}$$

$$= \log P(Y;\theta) - \sum_X q(X) \log \frac{q(X)}{P(X \mid Y;\theta)}$$

$$= \sum_X q(X) \log P(Y;\theta) + \sum_X q(X) \log P(X \mid Y;\theta) + H(q)$$

$$= \underbrace{\sum_X q(X) \log P(X,Y;\theta)}_{\text{M-step: } \theta \leftarrow \operatorname{argmax}} + H(q), \tag{2}$$

- We actually want to maximize $P(Y;\theta)$ w.r.t. $\theta \rightarrow$ but can't analytically
- Instead, we maximize the lower bound $F(q,\theta) \leq \log P(Y;\theta)$
  - E-step: find q that maximizes $F(q,\theta)$ for fixed θ^{old} using (1)
    ($\rightarrow q(X)$ approximates $P(X|Y;\theta)$, makes lower bound tight)
  - M-step: find θ that maximizes $F(q,\theta)$ for fixed q using (2)
    ($\rightarrow \theta$ maximizes $Q(\theta,q) = \sum_X q(X) \log P(X,Y;\theta)$)
- **Convergence proof:** $F(q,\theta)$ increases in each step.

- The free energy formulation of EM
  – clarifies convergence of EM (to *local minima*)
  – clarifies that q need only to *approximate* the posterior

- The free energy formulation of EM
  – clarifies convergence of EM (to *local minima*)
  – clarifies that q need only to *approximate* the posterior

- *Why is it called free energy?*

- The free energy formulation of EM
  – clarifies convergence of EM (to *local minima*)
  – clarifies that q need only to *approximate* the posterior

- *Why is it called free energy?*
  – Given a physical system in specific state (x, y), it has energy $E(x, y)$
    This implies a joint distribution $p(x, z) = \frac{1}{Z} e^{-E(x,y)}$ with partition function Z
  – If the DoFs y are "constrained" and x "unconstrained",
    then the *expected energy* is $\langle E \rangle = \sum_x q(x) E(x, y)$
    Note $\langle E \rangle = -Q(\theta, q)$ is the same as the neg. expected data log-likelihood
  – The unconstrained DoFs x have an entropy $H(q)$
  – Physicists call $A = \langle E \rangle - H$ (Helmholtz) free energy
    Which is the negative of our eq. (2): $F(q, \theta) = Q(\theta, q) + H(q)$

Conditional vs. generative models

(very briefly)

Maximum conditional likelihood

- We have two random variables X and Y and full data $D = \{(x_i, y_i)\}$ Given a model $P(X, Y; \theta)$ of the joint distribution we can train it

$$\theta^{\mathsf{ML}} = \operatorname*{argmax}_\theta \prod_i P(x_i, y_i; \theta)$$

- Assume the *actual goal* is to have a classifier $x \mapsto y$, or, the *conditional* distribution $P(Y|X)$.

  Q: Is it really necessary to first learn the full joint model $P(X, Y)$ when we only need $P(Y|X)$?

  Q: If $P(X)$ is very complicated but $P(Y|X)$ easy, then learning the joint $P(X, Y)$ is unnecessarily hard?

Maximum conditional Likelihood

Instead of likelihood maximization $\theta^{\mathsf{ML}} = \operatorname{argmax}_\theta \prod_i P(x_i, y_i; \theta)$:

- **Maximum conditional likelihood:**
  Given a *conditional model* $P(Y|X;\theta)$, train

$$\theta^* = \operatorname*{argmax}_\theta \prod_i P(y_i|x_i;\theta)$$

- The little but essential difference: We don't care to learn $P(X)$!

Example: Conditional Random Field

$$f(y,x) = \phi(y,x)^\top \beta \quad = \sum_{j=1}^{k} \phi_j(y_{\partial j}, x)\beta_j$$

$$p(y|x) = e^{f(y,x) - Z(x,\beta)} \quad \propto \prod_{j=1}^{k} e^{\phi_j(y_{\partial j}, x)\beta_j}$$

- "Random Field" $\leftrightarrow$ "Markov Random Field" $\leftrightarrow$ Factor Graph
  "Conditional" $\leftrightarrow$ we optimize for *conditional* likelihood.

- Logistic regression also maximizes the conditional log-likelihood
- Plain regression also maximizes the conditional log-likelihood! (with $\log P(y_i|x_i;\beta) \propto (y_i - \phi(x_i)^\top\beta)^2$)