Hidden Markov Models

Chris Williams, School of Informatics, University of Edinburgh

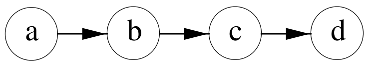
Overview

- Definitions
- Inference Problems
- Recursion formulae
- Viterbi alignment
- Training a HMM
- Linear-Gaussian HMMs (Kalman filtering)
- Reading: Jordan ch 12, Rabiner paper, Roweis paper

Dynamical models used in many areas for modelling sequences, including

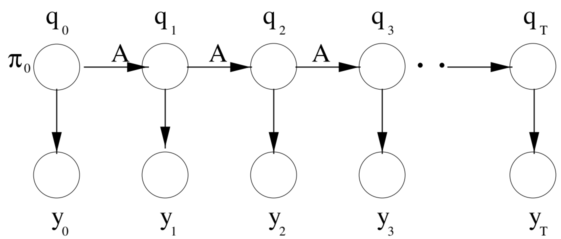
- Speech recognition
- Molecular biology sequences
- Linguistic sequences (e.g. part-of-speech tagging)
- Multi-electrode spike-train analysis
- Tracking objects through time

Markov Chain



$$P(a,b,c,d) = P(a)P(b|a)P(c|b)P(d|c)$$

Hidden Markov Model



A HMM is defined by

- M the number of states
- A the state transition matrix
- $P(y_t|q_t)$ the output probability distribution (independent of t)
- π the initial transition probabilities
- q_t is a multinomial variable with components q_t^i, if q_t is in state i then $q_t^i = 1$ and $q_t^j = 0$ for $j \neq i$

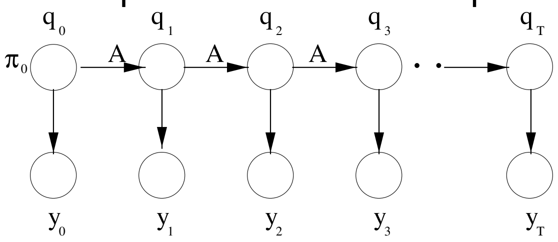
- Let $\pi_{q_0} = \prod_{i=1}^{M} [\pi_i]^{q_0^i}$ and

$$a_{q_t q_{t+1}} = \prod_{i,j} [a_{ij}]^{q_t^i q_{t+1}^j}$$

- Let $\mathbf{y} = y_0, y_1, \ldots, y_T$ and $\mathbf{q} = q_0, q_1, \ldots, q_T$
- For any state sequence $\mathbf{q}$

$$P(\mathbf{y}, \mathbf{q}) = \pi_{q_0} P(y_0|q_0) a_{q_0 q_1} P(y_1|q_1) \ldots a_{q_{T-1} q_T} P(y_T|q_T)$$

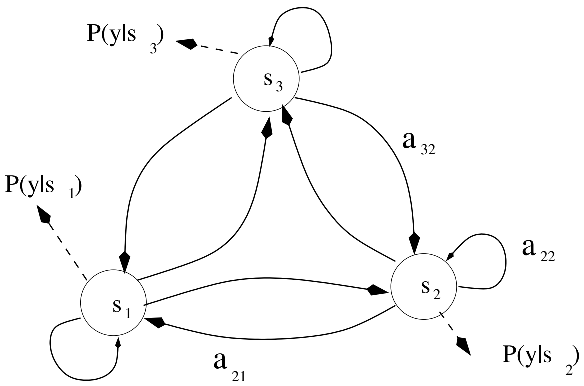
Independence relationships



- Conditioning on q_t renders q_{t-1} and q_{t+1} independent, i.e. $I(q_{t-1}, q_{t+1}|q_t)$
- $I(q_s, q_u|q_t)$ for all $s < t$, $u > t$

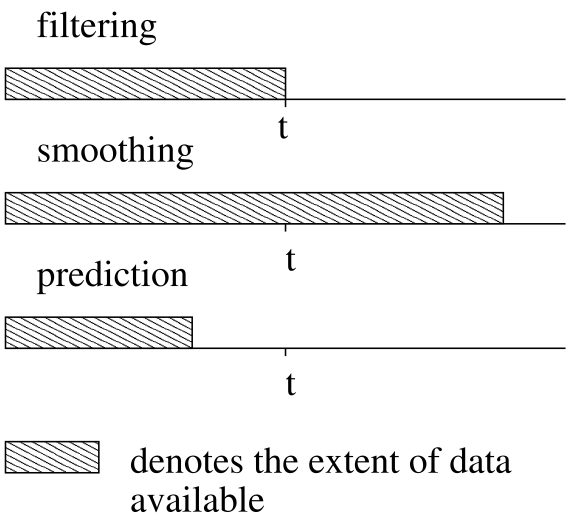
- $I(y_s, y_u|q_t)$ for all $s \leq t$, $u > t$
- the future is independent of the past given the present
- Note that conditioning on y_t does not yield any conditional independences
- HMM as a dynamical mixture model; choice of state is not independent at each time frame, but depends on the past

- HMM as a finite state automaton

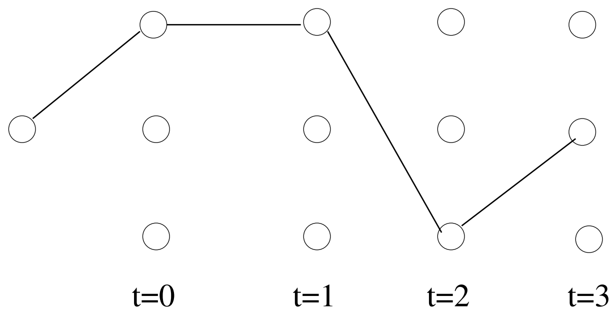


Inference Problems

- $P(q_0 \dots q_T|\mathbf{y})$ inferring hidden state given $\mathbf{y}$
- $P(q_t|\mathbf{y})$ marginal of above
- $P(q_t|y_0, \dots, y_t)$ filtering
- $P(q_t|y_0, \dots, y_s)$ $t > s$, prediction
- $P(q_t|y_0, \dots, y_u)$ $t < u$, smoothing
- $P(y_0, \dots, y_T)$ likelihood calculation
- Find sequence $q_0^* q_1^* \dots q_T^*$ that maximizes $P(\mathbf{q}|\mathbf{y})$ [Viterbi alignment]



$$P(y_0, \dots, y_T) = \sum_{q_0} \sum_{q_1} \dots \sum_{q_T} \pi(q_0) \prod_{t=0}^{T-1} a_{q_t q_{t+1}} \prod_{t=0}^{T} P(y_t|q_t)$$



- Naive approach $O(M^{T+1})$
- Efficient $O(M^2T)$ algorithm is available by pushing sums through products

$$\sum_{q_t} P(q_t|\mathbf{y}) = 1 = \frac{\sum_{q_t} \alpha(q_t)\beta(q_t)}{P(\mathbf{y})}$$

implies

$$P(\mathbf{y}) = \sum_{q_t} \alpha(q_t)\beta(q_t)$$

- Define $P(q_t|\mathbf{y}) = \gamma(q_t)$

Computing $P(q_t|\mathbf{y})$

$$\begin{aligned} P(q_t|\mathbf{y}) &= \frac{P(\mathbf{y}|q_t)P(q_t)}{P(\mathbf{y})} \\ &= \frac{P(y_0,\ldots,y_t|q_t)P(y_{t+1}\ldots y_T|q_t)P(q_t)}{P(\mathbf{y})} \\ &= \frac{P(y_0,\ldots,y_t,q_t)P(y_{t+1}\ldots y_T|q_t)}{P(\mathbf{y})} \\ &\equiv \frac{\alpha(q_t)\beta(q_t)}{P(\mathbf{y})} \end{aligned}$$

The alphas and betas can be calculated recursively.

Recursion formulae

- Alpha

  $$\alpha(q_{t+1}) = \sum_{q_t} \alpha(q_t) a_{q_t q_{t+1}} P(y_{t+1}|q_{t+1})$$

  Initialization

  $$\alpha(q_0) = P(y_0, q_0) = P(y_0|q_0)P(q_0) = P(y_0|q_0)\pi_{q_0}$$

- Beta

  $$\beta(q_t) = \sum_{q_{t+1}} \beta(q_{t+1}) a_{q_t q_{t+1}} P(y_{t+1}|q_{t+1})$$

  Initialization: $\beta(q_T)$ is the vector of ones as

  $$\sum_i \alpha(q_T^i)\beta(q_T^i) = \sum_i \alpha(q_T^i) = \sum_i P(y_0,\ldots,y_T,q_T^i) = P(\mathbf{y})$$

Each step is $O(M^2)$

Viterbi alignment

Find the best state sequence $q_0^* q_1^* \dots q_T^*$ that maximizes $P(\mathbf{q}|\mathbf{y})$

Define

$$\delta_t(i) = \max_{q_0, q_1, \dots q_{t-1}} P(q_0, q_1, \dots q_t = i, y_0 \dots y_t)$$

i.e. $\delta_t(i)$ is the best score along a single path up to time t which account for the first t observations and ends in state S_i.

There is a recursive formula for delta similar to the alpha recursion, except that a max rather than sum operation is used

For further details see, for example, L. Rabiner, Proc. IEEE 77(2) 1989 pp 257-285

Calculating $P(q_t, q_{t+1}|\mathbf{y})$

$$\begin{aligned}
\xi(q_t, q_{t+1}) &\equiv P(q_t, q_{t+1}|\mathbf{y}) \\
&= \frac{P(q_t, q_{t+1}, \mathbf{y})}{P(\mathbf{y})} \\
&= \frac{P(\mathbf{y}|q_t, q_{t+1}) P(q_{t+1}|q_t) P(q_t)}{P(\mathbf{y})} \\
&= P(y_0, \dots, y_t|q_t) P(y_{t+1}|q_{t+1}) \times \\
&\quad \frac{P(y_{t+2} \dots y_T|q_{t+1}) P(q_{t+1}|q_t) P(q_t)}{P(\mathbf{y})} \\
&= \frac{\alpha(q_t) P(y_{t+1}|q_{t+1}) \beta(q_{t+1}) a_{q_t q_{t+1}}}{P(\mathbf{y})}
\end{aligned}$$

Training a HMM

Use the EM algorithm to estimate π, A and η, the parameters of $P(y_t|q_t)$. Let $\theta = (\pi, A, \eta)$

If we knew the “true” state sequence, parameter estimation would be easy. The trick is to use the probability distribution over state paths to weight these estimates

$$\widehat{\pi}_i \leftarrow \gamma(q_0^i)$$

$$\widehat{a}_{ij} \leftarrow \frac{\sum_{t=0}^{T-1} \xi(q_t^i q_{t+1}^j)}{\sum_{t=0}^{T-1} \gamma(q_t^i)}$$

If the output is a multinomial distribution with $P(y_t^j = 1|q_t^i) = \eta_{ij}$ and $\sum_k y_t^k = 1$

$$\widehat{\eta}_{ij} \leftarrow \frac{\sum_{t=0}^{T} \gamma(q_t^i) y_t^j}{\sum_{t=0}^{T} \gamma(q_t^i)}$$

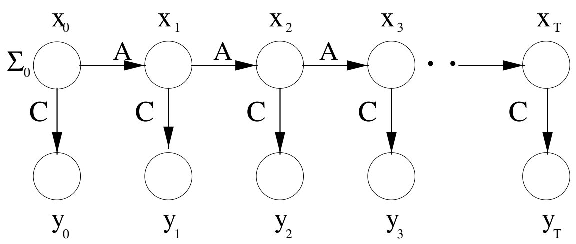
For HMMs these are known as the Baum-Welch equations

Example: Harmonizing Chorales in the Style of J S Bach

- Moray Allan and Chris Williams (NIPS 2004) `http://www.tardis.ed.ac.uk/~moray/harmony/`, online demo at `http://www.anc.inf.ed.ac.uk/demos/hmmbach/`
- Visible states are the melody (quarter notes)
- Hidden states are the harmony (which chord)
- Trained using labelled melody/harmony data (no need for EM)
- Task: find Viterbi alignment for harmony given melody (or sample from $P(\text{harmony}|\text{melody})$.)
- Actually uses HMMs for three subtasks: harmonic skeleton, chord skeleton, ornamentation

Linear-Gaussian HMMs

- Filtering problem known as Kalman filtering
- HMM with continuous state-space and observations



- Dynamical model
$$\mathbf{x}_{t+1} = A\mathbf{x}_t + G\mathbf{w}_t$$
where $\mathbf{w}_t \sim N(\mathbf{0}, Q)$ is Gaussian noise, i.e.
$$P(\mathbf{x}_{t+1}|\mathbf{x}_t) \sim N(A\mathbf{x}_t, GQG^T)$$

- Observation model
$$\mathbf{y}_t = C\mathbf{x}_t + \mathbf{v}_t$$
where $\mathbf{v}_t \sim N(\mathbf{0}, R)$ is Gaussian noise, i.e.
$$P(\mathbf{y}_t|\mathbf{x}_t) \sim N(C\mathbf{x}_t, R)$$

- Initialization
$$P(\mathbf{x}_0) \sim N(\mathbf{0}, \Sigma_0)$$

Inference Problem – filtering

- As whole model is Gaussian, only need to compute means and variances

$$P(\mathbf{x}_t|\mathbf{y}_0,\ldots,\mathbf{y}_t) \sim N(\hat{\mathbf{x}}_{t|t}, P_{t|t})$$

$$\hat{\mathbf{x}}_{t|t} = E[\mathbf{x}_t|\mathbf{y}_0,\ldots,\mathbf{y}_t]$$

$$P_{t|t} = E[(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t})(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t})^T|\mathbf{y}_0,\ldots,\mathbf{y}_t]$$

- Recursive update split into two parts
- **Time update**

$$P(\mathbf{x}_t|\mathbf{y}_0,\ldots,\mathbf{y}_t) \rightarrow P(\mathbf{x}_{t+1}|\mathbf{y}_0,\ldots,\mathbf{y}_t)$$

- **Measurement update**

$$P(\mathbf{x}_{t+1}|\mathbf{y}_0,\ldots,\mathbf{y}_t) \rightarrow P(\mathbf{x}_{t+1}|\mathbf{y}_0,\ldots,\mathbf{y}_t,\mathbf{y}_{t+1})$$

- Time update

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + G\mathbf{w}_t$$

thus

$$\hat{\mathbf{x}}_{t+1|t} = A\hat{\mathbf{x}}_{t|t}$$

$$P_{t+1|t} = AP_{t|t}A^T + GQG^T$$

- Measurement update (like posterior in Factor Analysis)

$$\hat{\mathbf{x}}_{t+1|t+1} = \hat{\mathbf{x}}_{t+1|t} + K_{t+1}(\mathbf{y}_{t+1} - C\hat{\mathbf{x}}_{t+1|t})$$

$$P_{t+1|t+1} = P_{t+1|t} - K_{t+1}CP_{t+1|t}$$

where

$$K_{t+1} = P_{t+1|t}C^T(CP_{t+1|t}C^T + R)^{-1}$$

K_{t+1} is known as the Kalman gain matrix

Simple example

$$x_{t+1} = x_t + w_t$$

$w_t \sim N(0,1)$

$$y_t = x_t + v_t$$

$v_t \sim N(0,1)$

$$P(x_0) \sim N(0,\sigma^2)$$

In the limit $\sigma^2 \rightarrow \infty$ we find

$$\hat{x}_{2|2} = \frac{5y_2 + 2y_1 + y_0}{8}$$

- Notice how later data has more weight
- Compare $x_{t+1} = x_t$ (so that w_t has zero variance); then

$$\hat{x}_{2|2} = \frac{y_2 + y_1 + y_0}{3}$$

Applications

Much as a coffee filter serves to keep undesirable grounds out of your morning mug, the Kalman filter is designed to strip unwanted noise out of a stream of data. *Barry Cipra, SIAM News 26(5) 1993*

- Navigational and guidance systems
- Radar tracking
- Sonar ranging
- Satellite orbit determination

Extensions

Dealing with non-linearity

- The Extended Kalman Filter (EKF)
  If $\mathbf{y}_t = f(\mathbf{x}_t) + \mathbf{v}_t$ where f is a non-linear function, can linearize f, e.g. around $\hat{\mathbf{x}}_{t|t-1}$. Works for weak non-linearities

- For very non-linear problems use sampling methods (known as particle filters). Example, work of Blake and Isard on tracking, see
  `http://www.robots.ox.ac.uk/~vdg/dynamics.html`

It is possible to train KFs using a forward-backward algorithm