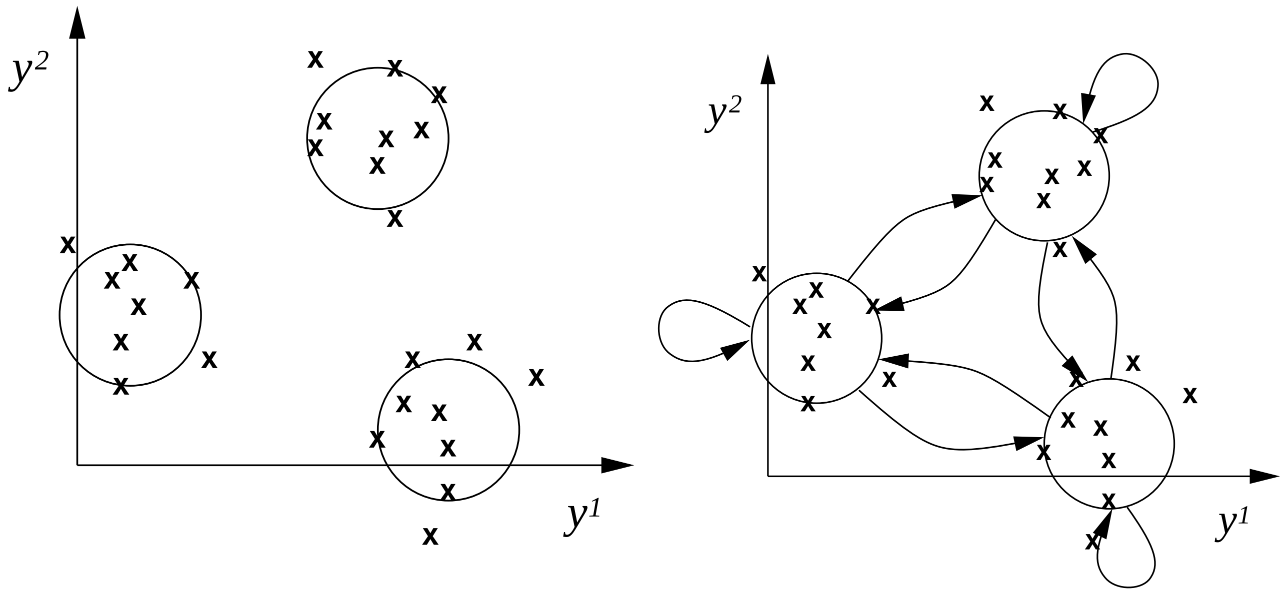
Uncertainty Modelling in AI

Hidden Markov Models

Lee Wee Sun

Semester 2, 2005

Hidden Markov Models (HMM)

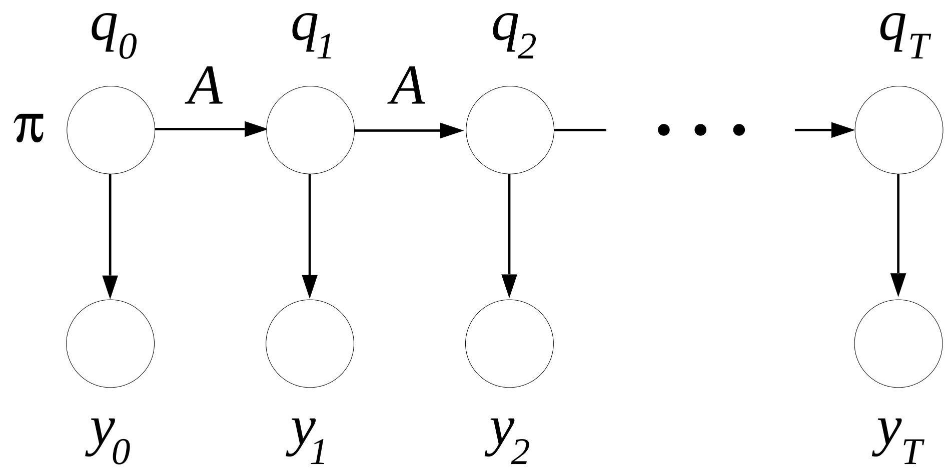


(a) (b)

- An HMM is a natural generalization of a mixture model
  - Can view as a "dynamic" mixture model.
- In a HMM, we call the components in a mixture the *states*.

- When we generate iid data under a mixture model (figure (a)),
  - At each step, a state is selected according to $p(z)$ independently of the choice of states of the other steps.
  - Given the state, a data vector is chosen from $p(x|z)$.
- With the HMM, instead to assuming that the states are chosen independently at each step, we assume that the choice of a state at a given step depends on the choice of the state at the previous step (figure (b), arrows representing transitions between states).
- If there are M states, then the transition probabilities can be represented by a $M \times M$ matrix, whose (i, j)-th entry represents the probability of transitioning from the i-th state at a given time to the j-th state at the next time step.

Graphical Model



- A HMM can be viewed as a linked sequence of mixture models.
- The state at time t is q_t and y_t is the "output" at time t.
- The random variable q_t is a multinomial random variable with components q_t^i, for $i = 1, \ldots, M$; q_t^i is equal to one for a particular value of i and is equal to zero for $j \neq i$.

- The output y_t is always observed in a HMM setting, thus we leave them unspecified but they may be considered as multinomial or Gaussian for concreteness.
- Conditioning on q_t renders q_{t+1} and q_{t-1} independent. In fact it renders q_s independent of q_u for $s < t$ and $t < u$. *The future is independent of the past, given the present.* Also true for y_s and y_u, given the state of q_t.
- Note that the HMM can also be considered as a tree.

Parameterization

- The first node has no parent, so it has an unconditional distribution π, where $\pi_i = p(q_0^i = 1)$.
- For each successive states, we have the previous state node in the chain as its parent, so require a $M \times M$ matrix to specify the conditional probability.
- Define a *state transition matrix* A, where the (i, j)-th entry a_{ij} of A is defined to be $p(q_{t+1}^j = 1 | q_t^i = 1)$.
- We assume that the transition probability is independent of t, i,e, a *homogeneous* HMM.
- Each output node has a single state node as parent, so we have $p(y_t | q_t)$. For the purpose of inference, we need only to be able to evaluate $p(y_t | q_t)$ for fixed y_t.

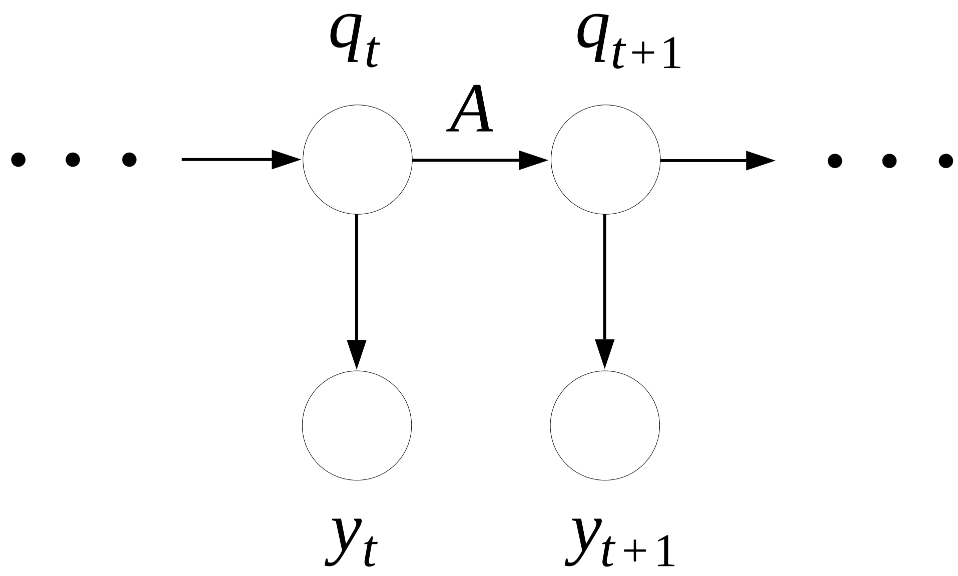
- For a particular configuration $(q, y) = (q_0, q_1, \dots, y_0, y_1, \dots, y_T)$, we obtain

$$p(q, y) = p(q_0) \prod_{t=0}^{T-1} p(q_{t+1}|q_t) \prod_{t=0}^{T} p(y_t|q_t).$$

- When q_t takes its i-th value and q_{t+1} takes its j-th value, we let $a_{q_t, q_{t+1}}$ denote the (i, j)-th entry of the matrix A. Similarly, when q_0 takes the i-th value let π_{q_0} denote π_i. Then we can write

$$p(q, y) = \pi_{q_0} \prod_{t=0}^{T-1} a_{q_t, q_{t+1}} \prod_{t=0}^{T} p(y_t|q_t).$$

Inference



- We take advantage of the conditional independencies in the graphical model to break the problem into pieces. Consider performing inference to obtain $\gamma(q_t) = p(q_t|y)$:

$$
\begin{aligned}
p(q_t|y) &= \frac{p(y|q_t)p(q_t)}{p(y)} \\
&= \frac{p(y_0, \ldots, y_t|q_t)p(y_{t+1}, \ldots, y_T|q_t)p(q_t)}{p(y)} && \text{cond. ind.} \\
&= \frac{p(y_0, \ldots, y_t, q_t)p(y_{t+1}, \ldots, y_T|q_t)}{p(y)} \\
&= \frac{\alpha(q_t)\beta(q_t)}{p(y)},
\end{aligned}
$$

where $\alpha(q_t)$ is the probability of emitting a partial sequence of outputs $y_0, \ldots, y_t$ and ending up in state q_t and $\beta(q_t)$ is the probability of emitting a partial sequence of outputs $y_{t+1}, \ldots, y_T$ given that the system starts in state q_t.

- The probability of the data $p(y)$ is easily calculated as the normalization factor

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

i.e. we can calculate $\alpha(q_t)$ and $\beta(q_t)$ for any t and sum their product.

- Exploiting conditional independence (see figure), $\alpha(q_t)$ and $\beta(q_t)$ can be calculated efficiently in a recursive manner.

$$
\begin{aligned}
\alpha(q_{t+1}) &= p(y_0, \ldots, y_{t+1}, q_{t+1}) \\
&= p(y_0, \ldots, y_{t+1} | q_{t+1}) p(q_{t+1}) \\
&= p(y_0, \ldots, y_t | q_{t+1}) p(y_{t+1} | q_{t+1}) p(q_{t+1}) \qquad \text{cond. ind.} \\
&= p(y_0, \ldots, y_t, q_{t+1}) p(y_{t+1} | q_{t+1}) \\
&= \sum_{q_t} p(y_0, \ldots, y_t, q_t, q_{t+1}) p(y_{t+1} | q_{t+1}) \\
&= \sum_{q_t} p(y_0, \ldots, y_t, q_{t+1} | q_t) p(q_t) p(y_{t+1} | q_{t+1}) \\
&= \sum_{q_t} p(y_0, \ldots, y_t | q_t) p(q_{t+1} | q_t) p(q_t) p(y_{t+1} | q_{t+1}) \qquad \text{c.i.} \\
&= \sum_{q_t} p(y_0, \ldots, y_t, q_t) p(q_{t+1} | q_t) p(y_{t+1} | q_{t+1})
\end{aligned}
$$

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

- Each step takes time $O(M^2)$ - for each of the M values of q_{t+1}, we need M multiplications.
- The values of $\alpha(q_t)$ is calculated "forward" from $t = 0$ to $t = T$. Total computation time for $t = 1$ to $t = T$ is $O(M^2 T)$.
- The initialization is

$$\begin{aligned} \alpha(q_0) &= p(y_0, q_0) \\ &= p(y_0|q_0)p(q_0) \\ &= p(y_0|q_0)\pi_{q_0}. \end{aligned}$$

- The values of $\beta(q_t)$ is calculated "backwards" from $t = T$ to $t = 0$.

$$
\begin{aligned}
\beta(q_t) &= p(y_{t+1}, \ldots, y_T | q_t) \\
&= \sum_{q_{t+1}} p(y_{t+1}, \ldots, y_T, q_{t+1} | q_t) \\
&= \sum_{q_{t+1}} p(y_{t+1}, \ldots, y_T | q_{t+1}, q_t) p(q_{t+1} | q_t) \\
&= \sum_{q_{t+1}} p(y_{t+2}, \ldots, y_T | q_{t+1}) p(y_{t+1} | q_{t+1}) p(q_{t+1} | q_t) \qquad \text{c.i.} \\
&= \sum_{q_{t+1}} \beta(q_{t+1}) a_{q_t, q_{t+1}} p(y_{t+1} | q_{t+1}).
\end{aligned}
$$

- The values of $\beta(q_T)$ is initialized to a vector of ones. We see that this gives the correct value for calculating $p(y)$.

$$
\begin{aligned}
p(y) &= \sum_i \alpha(q_T^i)\beta(q_T^i) \\
&= \sum_i \alpha(q_T^i) \\
&= \sum_i p(y_0, \ldots, y_T, q_T^i) \\
&= p(y).
\end{aligned}
$$

- Calculating $p(y)$ takes only a single pass. For any value of $\gamma(q_t) = p(q_t|y)$, $\alpha(q_i)$ only need to be calculated from $i = 0$ to $i = t$ while $\beta(q_i)$ only need to be calculated from $i = T$ to $i = t$ - single pass through the graphical model. To calculate all values of $\gamma(q_t)$ for $t = 0$ to $t = T$, we need to only do two passes through the graphical model.

- Another quantity of interest (particularly in the EM algorithm) is $\xi(q_t, q_{t+1}) = p(q_t, q_{t+1}|y)$.

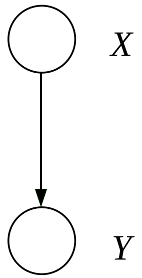
$$
\begin{aligned}
&\xi(q_t, q_{t+1}) \\
&= p(q_t, q_{t+1}|y) \\
&= \frac{p(y|q_t, q_{t+1})p(q_{t+1}|q_t)p(q_t)}{p(y)} \\
&= \frac{p(y_0, \ldots, y_t|q_t)p(y_{t+1}|q_{t+1})p(y_{t+2}, \ldots, y_T|q_{t+1})p(q_{t+1}|q_t)p(q_t)}{p(y)} \\
&= \frac{\alpha(q_t)p(y_{t+1}|q_{t+1})\beta(q_{t+1})a_{q_t,q_{t+1}}}{p(y)}
\end{aligned}
$$

- The quantity is easily calculated once the recursive calculations of the singleton probabilities $\gamma(q_t)$ are completed.

Maximum A Posteriori Probability for State Sequence

- We are often interested in finding the most probable configuration given the evidence rather than the conditional probabilities given the evidence.
  - For example, we may want to find the most probable word transmitted over a noisy channel given the received symbols. Finding the most probable character for each symbol is not the same as finding the most probable word.
  - In isolated word speech recognition, we want to maximize the probability of correction recognition of words. Spoken words are made up of a sequence of smaller speech units called phones. We want to maximize the probability of getting correct words instead of correct phones.

- Using the conditional probability for prediction minimizes the probability of incorrectly predicting the state. This is not the same as minimizing the probability of incorrectly predicting the whole sequence of states where the prediction is considered incorrect if any component state is incorrect.



1	.6
2	.2
3	.2

$p(x)$

	1	2	3
1	0	.6	.4
2	1	0	0
3	1	0	0

$p(y \mid x)$

1	2	3
.4	.36	.24

$p(y)$

- For example, in the figure above, the marginal probability of X is maximized by choosing $X = 1$ and the marginal for Y is maximized by choosing $Y = 1$. However, the configuration $(X = 1, Y = 1)$ has probability 0 while the maximizing assignment is $(X = 1, Y = 2)$.

Viterbi Algorithm

- By choosing $\gamma(q_t)$ that has the largest value for each t, we can find a sequence $q_0^*, q_1^*, \ldots, q_T^*$ with the *maximum expected number of correct states.*
- However, this may not be the sequence of states with the highest probability. In fact, it may be a sequence of probability zero (as seen previously).
- We are interested in finding the sequence that achieves $\max_{q_0,\ldots,q_T} p(q_0, \ldots, q_T | y_0, \ldots, y_T) =$ $\max_{q_0,\ldots,q_T} \frac{p(q_0,\ldots,q_T,y_0,\ldots,y_T)}{p(y_0,\ldots,y_T)}$.
- As the normalizer is the same for every sequence of states, this is the same as finding the sequence the sequence that achieves $\max_{q_0,\ldots,q_T} p(q_0, \ldots, q_T, y_0, \ldots, y_T)$.
- To find the best path, we use *Viterbi algorithm* which is has

the same recursion for calculating the $\alpha(q_t)$ except that $\sum$ is replaced with max.

- Here, instead of calculating $\alpha(q_{t+1}) = p(y_0, \ldots, y_{t+1}, q_{t+1})$, we calculate $\alpha'(q_{t+1}) = \max_{q_0,\ldots,q_t} p(q_0, \ldots, q_t, y_0, \ldots, y_{t+1}, q_{t+1})$.

$$\alpha'(q_{t+1}) = \max_{q_0,\ldots,q_t} p(q_0, \ldots, q_t, y_0, \ldots, y_{t+1}, q_{t+1})$$

$$= \max_{q_0,\ldots,q_t} p(q_0, \ldots, q_t, y_0, \ldots, y_t | q_{t+1}) p(y_{t+1}|q_{t+1}) p(q_{t+1}) \text{ c. i.}$$

$$= \max_{q_0,\ldots,q_t} p(q_0, \ldots, q_t, y_0, \ldots, y_t, q_{t+1}) p(y_{t+1}|q_{t+1})$$

$$= \max_{q_0,\ldots,q_t} p(q_0, \ldots, q_{t-1}, y_0, \ldots, y_t, q_{t+1}|q_t) p(q_t) p(y_{t+1}|q_{t+1})$$

$$= \max_{q_0,\ldots,q_t} p(q_0, \ldots, q_{t-1}, y_0, \ldots, y_t|q_t) p(q_{t+1}|q_t) p(q_t) p(y_{t+1}|q_{t+1}) \text{ c.i.}$$

$$= \max_{q_t} \max_{q_0,\ldots,q_{t-1}} p(q_0, \ldots, q_{t-1}, y_0, \ldots, y_t, q_t) p(q_{t+1}|q_t) p(y_{t+1}|q_{t+1})$$

$$= \max_{q_t} \alpha'(q_t) a_{q_t,q_{t+1}} p(y_{t+1}|q_{t+1})$$

- The initialization is $\alpha'(q_0) = p(y_0, q_0) = p(y_0|q_0)\pi_{q_0}$.

- The state q_T that maximizes $\alpha'(q_T)$ gives us the last state of the most probable sequence.
- Instead of calculating $\beta(q_t)$ backwards, we store the transitions that achieve the maximum for each state of each node on the forward pass and use a backward pass can to read off the rest of the most probable sequence.
- To do that, we use a table that stores the state from which the maximum value is achieved, $\delta(q_{t+1}) = \arg\max_{q_t} \alpha'(q_t) a_{q_t, q_{t+1}} p(y_{t+1}|q_{t+1})$, during the forward pass.
- In the backward pass, we start from the last state of the most probable sequence and follow the sequence of states pointed to by $\delta(q_t)$ backwards all the way to the first state of the sequence.

Example: Most Likely Word

Consider a language with 2 symbols a, b in the alphabet. The probabilities of generating the first character X_0 of a word is as follows: $p(X_0 = a) = 0.3$, $p(X_0 = b) = 0.7$. The probabilities of generating each of the following characters in the word is listed in the table below. Each row indicates the transition probabilities of going from the character indicated on the left of the row to the character indicated on the column.

	a	b
a	0.4	0.6
b	0.7	0.3

The observed characters are often corrupted by noise where the noise is independent for each character. For any character that is actually generated by the process, the probably of observing the correct character is 0.9.

- What is the probability of generating the word "baa"? Describe how it can be calculated efficiently for long words.

$$\begin{aligned} p(baa) &= p(b)p(a|b)p(a|a) \\ &= 0.7 * 0.7 * 0.4 \\ &= 0.196 \end{aligned}$$

This is a Markov model. Calculation can be done by simply looking up the conditional probabilities for each character conditioned on the previous character and multiplying the conditional probabilities together in time $O(n)$ where n is the number of characters.

- What is the probability of observing the word "baa"? Describe how it can be calculated efficiently for long words.

Let q_0, q_1 *and* q_2 *be the actual generated character and* y_0, y_1 *and* y_2 *be the observed characters. Then*
$p(baa) = \sum_{q_2 \in \{a,b\}} \alpha(q_2)$, *where* $\alpha(q_0) = p(b|q_0)p(q_0)$ *and*
$\alpha(q_{t+1}) = \sum_{q_t \in \{a,b\}} \alpha(q_t)p(q_{t+1}|q_t)p(y_{t+1}|q_{t+1})$. *This gives*

$$\alpha(q_0) = \begin{bmatrix} 0.03 \\ 0.63 \end{bmatrix} \alpha(q_1) = \begin{bmatrix} 0.4077 \\ 0.0207 \end{bmatrix} \alpha(q_2) = \begin{bmatrix} 0.159813 \\ 0.025083 \end{bmatrix}$$

$p(baa) = 0.184896$

The recursive formulas allow the probability of the observed word to be calculated efficiently in time $O(n)$.

- The observed word is “baa”. What is the probability that q_1 is actually b? Describe how it can be calculated efficiently for long words.

  We know that $p(q_1 = b|y) = \frac{\alpha(q_1=b)\beta(q_1=b)}{p(y)}$ *Need to calculate* $\beta(q_t = b) = \sum_{q_{t+1}\in\{a,b\}} \beta(q_{t+1})p(q_{t+1}|q_t)p(y_{t+1}|q_{t+1})$ *where* $\beta(q_t)$ *is initialized to vector of ones. This gives*

  $$\beta(q_1 = b) = 0.7 * 0.9 + 0.3 * 0.1 = 0.66.$$

  So $p(q_1 = b|y) = 0.66 * 0.0207/0.184896 = 0.074$. *The recursive formulas allow the probability to be calculated efficiently in time* $O(n)$.

- The observed word is “baa”. What is the most likely generated word. Describe how to calculated it efficiently for long words.

Recursively calculate
$\alpha'(q_{t+1}) = \max_{q_t} \alpha'(q_t) p(q_{t+1}|q_t) p(y_{t+1}|q_{t+1})$. *Also, store a table* $\delta(q_{t+1}) = \arg\max_{q_t} \alpha(q_t) p(q_{t+1}|q_t) p(y_{t+1}|q_{t+1})$. *Use this table to trace backwards from the last state* q_t *to find the most likely word. This gives*

$$\alpha(q_0) = \begin{bmatrix} 0.03 \\ 0.63 \end{bmatrix} \qquad \alpha(q_1) = \begin{bmatrix} 0.3969 \\ 0.0189 \end{bmatrix} \qquad \delta(q_1) = \begin{bmatrix} b \\ b \end{bmatrix}$$

$$\alpha(q_2) = \begin{bmatrix} 0.142884 \\ 0.023814 \end{bmatrix} \qquad \delta(q_2) = \begin{bmatrix} a \\ a \end{bmatrix}$$

Tracing back gives the most likely generated word as “baa” with probability 0.142884. The recursive formulas allow the most likely word to be calculated efficiently in time $O(n)$.

Parameter Estimation

- We consider parameter estimation for one sequence y of observations. Estimation for multiple independent observation sequences requires only minor adjustment to the algorithm.
- We start with the complete log likelihood. If the states are observed, this would provide the final estimate.
- For concreteness, we derive the algorithm for the case in which the outputs y_t are multinomial random variables.
- For multinomials, y_t is an N-component vector such that y_t^j is one for one component and zero for all other components.

- Let η_{ij} denote the probability that the j-th component of y_t is one, given that the i-th component of q_t is one, $\eta_{ij} = p(y_i^j = 1 | q_t^i = 1, \eta)$. We can write

$$p(y_t | q_t, \eta) = \prod_{i,j=1}^{M,N} \eta_{ij}^{q_t^i y_t^j}$$

as the general expression for the output distribution.

- The complete log likelihood

$$
\begin{aligned}
&\log p(q, y) \\
&= \log \left\{ \pi_{q_0} \prod_{t=0}^{T-1} a_{q_t, q_{t+1}} \prod_{t=0}^{T} p(y_t | q_t, \eta) \right\} \\
&= \log \left\{ \prod_{i=1}^{M} \pi_i^{q_0^i} \prod_{t=0}^{T-1} \prod_{i,j=1}^{M} a_{ij}^{q_t^i, q_{t+1}^j} \prod_{t=0}^{T} \prod_{i,j=1}^{M,N} \eta_{ij}^{q_t^i y_t^j} \right\} \\
&= \sum_{i=1}^{M} q_0^i \log \pi_i + \sum_{t=0}^{T-1} \sum_{i,j=1}^{M} q_t^i, q_{t+1}^j \log a_{ij} \sum_{t=0}^{T} \sum_{i,j=1}^{M,N} q_t^i y_t^j \log \eta_{ij}
\end{aligned}
$$

- The maximum likelihood estimate for the case of complete data:

$$\begin{aligned}
\hat{a}_{ij} &= \frac{m_{ij}}{\sum_{k=1}^{M} m_{ik}} \\
\hat{\eta}_{ij} &= \frac{n_{ij}}{\sum_{k=1}^{N} n_{ik}} \\
\hat{\pi}_i &= q_0^i,
\end{aligned}$$

  where the sufficient statistics $m_{ij} = \sum_{t=0}^{T-1} q_t^i q_{t+1}^j$ and $n_{ij} = \sum_{t=0}^{T} q_t^i y_t^j$.

- Note that m_{ij} is the number of times that the process is in state i and transitions to state j. Dividing by $\sum_k m_{ik}$ gives the proportion of transitions from state i that goes to state j. Similarly η_{ij} is the proportion of times that the chain is in state i and produces the j-th output value.

- When the data is not fully observed, we can use the EM algorithm.
- In the E step of EM, we calculate the expected value of the sufficient statistics $n_{ij} = \sum_{t=0}^{T} q_t^i y_t^j$

$$
\begin{aligned}
E(n_{ij}|y, \theta^{(p)}) &= \sum_{t=0}^{T} E(q_t^i|y, \theta^{(p)}) y_t^j \\
&= \sum_{t=0}^{T} p(q_t^i = 1|y, \theta^{(p)}) y_t^j \\
&= \sum_{t=0}^{T} \gamma_t^{i(p)} y_t^j .
\end{aligned}
$$

- Similarly for $m_{ij} = \sum_{t=0}^{T-1} q_t^i q_{t+1}^j$

$$\begin{aligned} E(m_{ij}|y, \theta^{(p)}) &= \sum_{t=0}^{T-1} E(q_t^i q_{t+1}^j | y, \theta^{(p)}) \\ &= \sum_{t=0}^{T-1} p(q_t^i q_{t+1}^j | y, \theta^{(p)}) \\ &= \sum_{t=0}^{T-1} \xi_{t,t+1}^{ij(p)}. \end{aligned}$$

- Substituting into the maximum likelihood formulas, we get

$$\hat{\eta}_{ij}^{(p+1)} = \frac{\sum_{t=0}^{T} \gamma_t^{i(p)} y_t^j}{\sum_{k=1}^{N} \sum_{t=0}^{T} \gamma_t^{i(p)} y_t^k} = \frac{\sum_{t=0}^{T} \gamma_t^{i(p)} y_t^j}{\sum_{t=0}^{T} \gamma_t^{i(p)}},$$

where we used $\sum_{k=1}^{N} y_t^k = 1$, and

$$\hat{a}_{ij}^{(p+1)} = \frac{\sum_{t=0}^{T-1} \xi_{t,t+1}^{ij(p)}}{\sum_{k=1}^{M} \sum_{t=0}^{T-1} \xi_{t,t+1}^{ik(p)}} = \frac{\sum_{t=0}^{T-1} \xi_{t,t+1}^{ij(p)}}{\sum_{t=0}^{T-1} \gamma_t^{i(p)}},$$

where $\sum_{k=1}^{M} \xi_{t,t+1}^{ik(p)} = \gamma_t^{i(p)}$, and

$$\hat{\pi}_i^{(p+1)} = \gamma_0^{i(p)}.$$