Hidden Markov models

ELEM041 - Machine Learning

March 12, 2009

In these notes we will go through the definitions and derivations for Markov chains and hidden Markov models (HMMs). First the concept of conditional independence is introduced in § 1, then Markov chains and learning in Markov chains are covered in § 2 and § 3; HMMs are defined in § 4, with methods for inference and learning described in § 5 and § 6.

1 Conditional independence

Two events A and B are *conditionally independent* given third event C if and only if $P(A \cap B|C) = P(A|C)P(B|C)$. Note that this is the same as ordinary independence in a probability model where everything is conditioned on C. Like ordinary pairwise independence, this statement can also be several equivalent ways:

$$P(A|B \cap C) = P(A|C) \quad \text{and} \quad P(B|A \cap C) = P(B|C) \tag{1}$$

Two random variables $X : \Omega \to \mathbb{R}$ and $Y : \Omega \to \mathbb{R}$ are *conditionally independent* given third $Z : \Omega \to \mathbb{R}$ if and only if

$$\forall x, y, z \in \mathbb{R},\ P(X < x \cap Y < y | Z < z) = P(X < x | Z < z) P(Y < y | Z < z). \tag{2}$$

This is sometimes written as $X \perp Y|Z$. For multivariate random variables, the three equivalent statements of conditional independence can be written in terms of probability density functions:

$$\begin{aligned} p_{X,Y|Z}(x, y|z) &= p_{X|Z}(x|z)p_{Y|Z}(y|z) \\ p_{X|Y,Z}(x|y, z) &= p_{X|Z}(x|z) \\ p_{Y|X,Z}(y|x, z) &= p_{X|Z}(y|z) \end{aligned} \tag{3}$$

Conditional independencies in a probabilistic model mean that the full joint probability distribution of all the variables factorises, which tends to make the model more tractable. For example, if $X \perp Y|Z$, then

$$p(x, y, z) = p(x, y|z)p(z) = p(x|z)p(y|z)p(z),$$

or alternatively,

$$p(x, y, z) = p(x|y, z)p(y, z) = p(x|z)p(z|y)p(y).$$

Notice the way that the variables form a chain in the last expression.

2 Markov chains

A Markov chain is a sequence of random variables $S_t : \Omega \to \mathcal{S}$, for $t \in \mathbb{N}$, where each element of the sequence in conditionally independent of all the others *given* its predecessor, that is $S_{t+1} \perp S_u | S_t$ for all $u \notin \{t, t+1\}$. The probability distribution factorises into a product of *transition probabilities* involving neighbouring variables. For example, if $\mathbf{s} = (s_1, \ldots, s_T)$ is a sequence sampled from the Markov chain (the entire sequence is *one* sample), then the probability distribution is

$$p_{\mathbf{S}}(\mathbf{s}) = p_{S_1}(s_1) \prod_{t=1}^{T-1} p_{S_{t+1}|S_t}(s_{t+1}|s_t). \tag{4}$$

The individual factors may be density functions or mass functions depending on whether $\mathcal{S}$ is a continuous or discrete domain respectively.

To continue, we will assume that we are dealing with a discrete-valued Markov chain, with $\mathcal{S} = \{1, \ldots, K\}$, (or $1..K$ for short), and that the model is time-shift invariant, which means that the conditional probabilities $p_{S_{t+1}|S_t}(s_{t+1}|s_t)$ depend only on the values s_{t+1} and s_t and not on t itself. In this case, the model can be parameterised in terms of $\pi \in \mathbb{R}^K$, the probability distribution for the first element of the chain S_1, and $a \in \mathbb{R}^{K \times K}$, the transition probability matrix:

$$\pi_k = P(S_1 = k), \qquad a_{jk} = P(S_{t+1} = k | S_t = j). \tag{5}$$

These parameters must yield properly normalised distributions, so $\sum_{k=1}^{K} \pi_k = 1$ and $\forall j \in 1..K$, $\sum_{k=1}^{K} a_{jk} = 1$. In terms of the these, the probability of a given sequence $\mathbf{s}$ is

$$p_{\mathbf{S}}(\mathbf{s}; \pi, a) = \pi_{s_1} \prod_{t=1}^{T-1} a_{s_t, s_{t+1}}. \tag{6}$$

3 ML learning of Markov chain parameters

The parameters of a Markov chain can be estimated from an observed sequence by maximising the log likelihood

$$\mathcal{L}(\pi, a) = \log p_{\mathbf{S}}(\mathbf{s}; \pi, a) = \log \pi_{s_1} + \sum_{t=1}^{T-1} \log a_{s_t, s_{t+1}}. \tag{7}$$

This can also be written as

$$\mathcal{L}(\pi, a) = \log \pi_{s_1} + \sum_{j=1}^{K} \sum_{k=1}^{K} \left(\sum_{t=1}^{T-1} \delta_{j, s_t} \delta_{k, s_{t+1}} \right) \log a_{jk}. \tag{8}$$

Notice how the data dependence is entirely contained in the sum in brackets and does not involve the parameters. In fact $N_{jk} = \sum_{t=1}^{T-1} \delta_{j, s_t} \delta_{k, s_{t+1}}$ is essentially the 2-dimensional histogram of transitions in the sequence.

To optimise $\mathcal{L}(\pi, a)$ under the $K+1$ normalisation constraints we use the method of Lagrange multipliers and find the stationary points of:

$$\mathcal{L}^{\lambda}(\pi, a) = \log \pi_{s_1} + \left[\sum_{j=1}^{K}\sum_{k=1}^{K} N_{jk} \log a_{jk}\right] - \left[\lambda^{\pi} \sum_{k=1}^{K} \pi_k\right] - \left[\sum_{j=1}^{K} \lambda_j \sum_{k=1}^{K} a_{jk}\right], \quad (9)$$

where λ^{π} and $\lambda_1, \ldots, \lambda_k$ are the Lagrange multipliers. Partial differentiation with respect to the π_k yields

$$\begin{aligned}\frac{\partial \mathcal{L}^{\lambda}(\pi, a)}{\partial \pi_i} &= \frac{1}{\pi_{s_1}} \frac{\partial \pi_{s_1}}{\partial \pi_i} - \lambda^{\pi} \sum_{k=1}^{K} \frac{\partial \pi_k}{\partial \pi_i} \\ &= \frac{1}{\pi_i} \delta_{i,s_1} - \lambda^{\pi} \sum_{k=1}^{K} \delta_{k,i} = \frac{1}{\pi_i} \delta_{i,s_1} - \lambda^{\pi}.\end{aligned}$$

Setting this to zero and using the constraint gives

$$\hat{\pi}_i = \frac{\delta_{i,s_1}}{\lambda^{\pi}} \quad \Longrightarrow \quad \sum_{i=1}^{K} \hat{\pi}_i = \frac{1}{\lambda} = 1 \quad \Longrightarrow \quad \hat{\pi}_i = \delta_{i,s_1}. \quad (10)$$

So the ML estimate $\hat{\pi}$ is just a distribution with all of its mass on whatever happens to be the first value in the sequence. This is clearly not a very insightful piece of inference, but since the data consists of only one instance of the Markov chain, it is the best that can be done. It is possible to repeat the analysis using multiple sampled sequences, in which case, the ML estimate of π will be the normalised histogram of starting values.

Similarly, partial differentation with resect to the a_{jk} yields

$$\frac{\partial \mathcal{L}^{\lambda}(\pi, a)}{\partial a_{jk}} = \frac{N_{jk}}{a_{jk}} - \lambda_j \quad \Longrightarrow \quad \hat{a}_{jk} = \frac{N_{jk}}{\lambda_j}.$$

The normalisation constraints give

$$\sum_{k=1}^{K} \hat{a}_{jk} = \sum_{k=1}^{K} \frac{N_{jk}}{\lambda_j} = 1 \implies \lambda_j = \sum_{k=1}^{K} N_{jk} \implies \hat{a}_{jk} = \frac{N_{jk}}{\sum_{i=1}^{K} N_{ji}}.$$

Hence, the estimated transition matrix is the row-wise normalisation of the histogram of observed transitions. So, to summarise, the maximum likelihood estimates of the parameters π and a are

$$\hat{\pi}_k = \delta_{k,s_1}, \qquad \hat{a}_{jk} = \frac{N_{jk}}{\sum_{i=1}^{K} N_{ji}}, \qquad N_{jk} = \sum_{t=1}^{T-1} \delta_{j,s_t} \delta_{k,s_{t+1}} \quad (11)$$

4 Hidden Markov models

A hidden Markov model consists of two sequences of random variables $S_1, S_2, \ldots$ and $X_1, X_2, \ldots$ where the S_t are a Markov chain as defined above and, for each $t \in \mathbb{N}$, $X_t : \Omega \to \mathcal{X}$ is random variable which is dependent on S_t but conditionally independent of every other variable in the model *given* S_t. This means that the joint probability of the two sequences $\mathbf{s} = (s_1, \ldots, s_T)$ and $\mathbf{x} = (x_1, \ldots, x_T)$ factorises as

$$p_{\mathbf{X},\mathbf{S}}(\mathbf{x}, \mathbf{s}) = p_{\mathbf{S}}(\mathbf{s}) \prod_{t=1}^{T} p_{X_t|S_t}(x_t|s_t) \tag{12}$$

$$= p_{S_1}(s_1) \prod_{t=1}^{T-1} p_{S_{t+1}|S_t}(s_{t+1}|s_t) \prod_{t=1}^{T} p_{X_t|S_t}(x_t|s_t). \tag{13}$$

We'll assume as before that the state space of the Markov chain is discrete, consisting of the integers 1 to K, and that the model is time-shift invariant, in which case the conditional distributions $p_{X_t|S_t}$, sometimes called emission or observation probabilities, are all the same function independent of t. Since the emission distribution is conditioned on S_t, which can take one of K values, it amounts to collection of K state-conditional distributions which we can write as f_k. Putting all of this together, we obtain the following expression for the joint probability:

$$p_{\mathbf{X},\mathbf{S}}(\mathbf{x}, \mathbf{s}) = \pi_{s_1} \prod_{t=1}^{T-1} a_{s_t, s_{t+1}} \prod_{t=1}^{T} f_{s_t}(x_t). \tag{14}$$

The most appropriate form and parameterisation of the state conditional distributions f_k depends on whether the observtion domain $\mathcal{X}$ is discrete or continuous and whether or not the K distributions are from the same family.

Discrete output HMMs If the observation domain $\mathcal{X}$ is the set of integers $\{1, \ldots, N\}$, then the state conditional distributions f_k amount to K discrete distributions over N values and can parameterised in terms of a matrix $b \in \mathbb{R}^{K \times N}$:

$$b_{ji} = P(X_t{=}i|S_t{=}j), \qquad \sum_{i=1}^{N} b_{ji} = 1. \tag{15}$$

Emission distributions from a parameterised family If the state conditional distributions are all from the same parameterised family $\mathcal{M}$, then we can introduce parameters $\phi_k \in \Phi_{\mathcal{M}}, k \in \{1, ..., K\}$, where $\Phi_{\mathcal{M}}$ is the appropriate parameter space for that family. In this case, we can define

$$f_k(x) = f_{\mathcal{M}}(x; \phi_k), \tag{16}$$

where $f_{\mathcal{M}} : \mathcal{X} \times \Phi_{\mathcal{M}} \to \mathbb{R}$ is now a single parametric probability density punction. For example, if $\mathcal{X} = \mathbb{R}^m$ and the emission distributions are all multivariate normal (denoted by $\mathcal{N}$), then they can be parameterised in the usual way by K means

$\mu^{(k)} \in \mathbb{R}^m$ and K covariance matrices $C^{(k)}$. In this case, $\phi_k = (\mu^{(k)}, C^{(k)})$, the parameter space is $\Phi_{\mathcal{N}} = \mathbb{R}^m \times \mathbb{R}^{m\times m}$ and

$$f_{\mathcal{N}}(x;\mu,C) = \frac{1}{\sqrt{|C|(2\pi)^m}} \exp -\tfrac{1}{2}(x-\mu)^{\mathrm{T}} C^{-1}(x-\mu). \tag{17}$$

Note the similarity between the observation part of the model and a Gaussian mixture model. In any case, the fully parameterised and factorised form of the joint distribution is now

$$p_{\mathbf{X},\mathbf{S}}(\mathbf{x},\mathbf{s}) = \pi_{s_1} \prod_{t=1}^{T-1} a_{s_t,s_{t+1}} \prod_{t=1}^{T} f_{\mathcal{M}}(x_t;\phi_{s_t}). \tag{18}$$

5 Inference

In applications, it is usually the case that values of the variables X_t are observed over some range of t and the corresponding values of S_t are unknown. The term 'inference' here covers various operations on the posterior distribution of the hidden state sequence $\mathbf{s} = (s_1,\ldots,s_T)$ given the observed sequence $\mathbf{x} = (x_1,\ldots,x_T)$ and the parameters of the model; that is, computations on

$$p_{\mathbf{S}|\mathbf{X}}(\mathbf{s}|\mathbf{x}) = \frac{p_{\mathbf{X},\mathbf{S}}(\mathbf{x},\mathbf{s})}{p_{\mathbf{X}}(\mathbf{x})} \tag{19}$$

$$\text{with} \quad p_{\mathbf{X}}(\mathbf{x}) = \sum_{\mathbf{s}\in(1..K)^T} p_{\mathbf{X},\mathbf{S}}(\mathbf{x},\mathbf{s}). \tag{20}$$

The problem with this is the size of the space of possible state sequences, of which there are K^T, and so we immediately have a problem in computing $p_{\mathbf{X}}(\mathbf{x})$ given an observed sequence. We will see that the conditional independence structure of the model means that efficient solutions to this and related inference tasks are available.

In the following, we will assume that the observation space $\mathcal{X}$ is the discrete set $1..N$, which means that the we can describe observations in terms of events such as $X_t = x_t$ instead of probability density functions, but the end results generalise in obvious ways to continuous observation spaces and their associated density functions. We will sometimes refer to subsequences of state or observation variables and their values as, e.g. $X_{a:b} = (X_a, X_{a+1}, \ldots, X_b)$ or $x_{a:b} = (x_a,\ldots,x_b)$. This gives us a convenient notation for events such as $(X_{1:t} = x_{1:t})$ which is equivalent to the intersection of several events $\bigcap_{i=1}^{t}(X_i = x_i)$. In fact, this particular class of *observation* event is so useful we adopt an even more succinct notation, $O_a^b \equiv (X_{a:b} = x_{a:b})$.

5.1 Filtering, predicition and smoothing

An HMM is an example of a probabilistic model whos variables can correspond to a temporal sequence. In applications, the observations $x_1, x_2, \ldots$ may arrive in real time, one by one, in which case it becomes relevant to consider what kinds of inference can be done at time t using only the data that has arrived so far. There are a few common scenarios (the names come from typical applications in signal processing and source coding):

Filtering involves the posterior marginal distribution of the 'current' state S_t given all the data up till now, $x_{1:t}$. In the HMM, this corresponds to computing $P(S_t = k|O_1^t)$ for all $k \in 1..K$. Once this is done, an application might take the posterior expectation E $S_t|O_1^t$, the posterior mode $\arg\max_k P(S_t = k|O_1^t)$, or some other decision on the basis of posterior distribution.

Fixed lag smoothing involves computing the posterior marginal distribution of the state variable S_t only after a certain lag τ. In the HMM, this corresponds to $P(S_t = k|O_1^{t+\tau})$. The extra τ observations $x_{t+1:t+\tau}$ following time t can often reduce our uncertainty about S_t.

Fixed interval smoothing involves computing all the posterior marginals only after all the observations have arrived; that is computing $P(S_t = k|O_1^T)$ for all $k \in 1..K$ and $t \in 1..T$.

Prediction involves computing the posterior marginals for states in the future, that is, $P(S_{t+\tau} = k|O_1^t)$ for some $\tau > 0$.

Decoding involves finding the maximum *a posteriori* estimate of the entire hidden state sequence given all the data, that is $\arg\max_{\mathbf{s}} p_{\mathbf{S}|\mathbf{X}}(\mathbf{s}|\mathbf{x})$, or, in terms of event probablities, $\arg\max_{\mathbf{s}} P(S_{1:t} = s_{1:t}|O_1^t)$.

5.2 Forwards algorithm

The forwards algorithm provides an efficient way to compute the marginal probability $p_{\mathbf{X}}(\mathbf{x}) = P(O_1^T)$ as well as the filtering distribution $P(S_t = k|O_1^t)$. It does so by exploiting a recursive expression for $P(S_t = k \cap O_1^t)$

Consider the four events O_1^{t-1}, $(S_{t-1} = j)$, $(S_t = k)$, and $O_t^t \equiv (X_t = x_t)$. The conditional independence structure of the HMM means that S_t is independent of everything given X_t and S_t is independent of everything else given S_{t-1}, so the joint probability of the four events is

$$\begin{aligned}
&P(O_t^t \cap S_t = k \cap S_{t-1} = j \cap O_1^{t-1} \cap) \\
&= P(O_t^t|S_t = k)P(S_t = k|S_{t-1} = j \cap O_1^{t-1})P(S_{t-1} = j \cap O_1^{t-1}) \quad [\text{since } X_t \perp \bullet|S_t] \\
&= P(O_t^t|S_t = k)P(S_t = k|S_{t-1} = j)P(S_{t-1} = j \cap O_1^{t-1}) \quad [\text{since } S_t \perp \bullet|S_{t-1}].
\end{aligned}$$

Since the events $S_{t-1} = j$ for $j \in 1..K$ are mutually exclusive and exhaustive, the total probability theorem applies and, noting that $O_1^t \equiv (O_1^{t-1} \cap O_t^t)$, we obtain

$$\begin{aligned}
P(O_1^t \cap S_t = k) &= \sum_{j=1}^{K} P(O_1^{t-1} \cap O_t^t \cap S_t = k \cap S_{t-1} = j) \\
&= \sum_{j=1}^{K} P(O_t^t|S_t = k)P(S_t = k|S_{t-1} = j)P(S_{t-1} = j \cap O_1^{t-1}).
\end{aligned}$$

The middle factor $P(S_t\!=\!k|S_{t-1}\!=\!j)$ is just the transition probability a_{jk}, and we define α and B to abbreviate the other two:

$$\alpha_k^{(t)} \overset{\text{def}}{=} P(O_1^t \cap S_t\!=\!k) \tag{21}$$

$$B_k^{(t)} \overset{\text{def}}{=} P(X_t\!=\!x_t|S_t\!=\!k). \tag{22}$$

In terms of these definitions, we get the forward recurrence relation and its initialisation for $t = 1$:

$$\alpha_k^{(t)} = B_k^{(t)} \sum_{j=1}^{K} a_{jk}\alpha_j^{(t-1)}, \qquad \alpha_k^{(1)} = B_k^{(1)}\pi_k, \tag{23}$$

where the latter comes from the definition of $\alpha_k^{(t)}$ at $t = 1$:

$$P(S_1\!=\!k \cap O_1^1) = P(X_1\!=\!x_1|S_1\!=\!k)P(S_1\!=\!k). \tag{24}$$

Given the values of $\alpha_k^{(t)}$ for $k \in 1..K$ at any time t, we can compute the probability $P(O_1^t)$ of all the data observed so far using the total probability theorem:

$$P(O_1^t) = \sum_{k=1}^{K} P(O_1^t \cap S_t\!=\!k) = \sum_{k=1}^{K} \alpha_k^{(t)}. \tag{25}$$

In particular, we can compute the marginal probability $p_{\mathbf{X}}(\mathbf{x})$ of the entire sequence since it is $P(O_1^T) = \sum_{k=1}^{K} \alpha_k^{(T)}$. We can also compute the filtering distribution

$$P(S_t\!=\!k|O_1^t) = \frac{P(S_t\!=\!k \cap O_1^t)}{P(O_1^t)} = \frac{\alpha_k^{(t)}}{\sum_{j=1}^{K} \alpha_j^{(t)}}. \tag{26}$$

5.3 Backwards algorithm

The backwards algorithm also enables us to compute the probability of the observed sequence $P(O_1^T)$, and in combination with the forwards algorithm, the smoothed posterior marginals $P(S_t = k|O_1^T)$ for $t < T$. It does so by exploiting a recursive expression for $P(O_{t+1}^T|S_t\!=\!k)$

Consider the four events $(S_{t-1} = j)$, $(S_t = k)$, $(X_t = x_t)$ and O_{t+1}^T. The joint probability of the latter three conditioned on the first can be written in factorised form by exploiting conditional independence as we did with for forwards algorithm:

$$\begin{aligned} & P(O_{t+1}^T \cap O_t^t \cap S_t\!=\!k|S_{t-1}\!=\!j) \\ &= P(O_{t+1}^T \cap O_t^t|S_t\!=\!k)P(S_t\!=\!k|S_{t-1}\!=\!j) \quad [\text{since } X_{t:T} \perp \bullet|S_t] \\ &= P(O_{t+1}^T|S_t\!=\!k)P(O_t^t|S_t\!=\!k)P(S_t\!=\!k|S_{t-1}\!=\!j) \quad [\text{since } X_t \perp \bullet|S_t]. \end{aligned}$$

We again use the total probability theorem, summing over the events $S_t\!=\!k$:

$$\begin{aligned} P(O_t^T|S_{t-1}\!=\!j) &= \sum_{k=1}^{K} P(O_{t+1}^T \cap O_t^t \cap S_t\!=\!k|S_{t-1}\!=\!j) \\ &= \sum_{k=1}^{K} P(O_{t+1}^T|S_t\!=\!k)P(O_t^t|S_t\!=\!k)P(S_t\!=\!k|S_{t-1}\!=\!j). \end{aligned}$$

We define $\beta_j^{(t)}$ as the conditional probability of the data after t given the state at t:

$$\beta_j^{(t)} \stackrel{\text{def}}{=} P(O_{t+1}^T | S_t = j), \tag{27}$$

in terms of which we obtain the backward recurrence relation and initialistion:

$$\beta_j^{(t-1)} = \sum_{k=1}^K \beta_k^{(t)} B_k^{(t)} a_{jk}, \qquad \beta_j^{(T)} = 1. \tag{28}$$

Given the values of $\beta_k^{(1)}$ for $k \in 1..K$, we can derive, using the same kind of reasoning as we did to get the backwards recurrence relation, the probability of all the data:

$$\begin{aligned} P(O_1^T) &= \sum_{k=1}^K P(O_2^T | S_1 = k) P(O_1^1 | S_1 = k) P(S_1 = k) \\ &= \sum_{k=1}^K \beta_k^{(1)} B_k^{(1)} \pi_k. \end{aligned} \tag{29}$$

However, the backwards algorithm is not normally used for this purpose, but rather in combination with the results of the forwards algorithm.

5.4 Forwards-backwards algorithm

By combining the results of the forwards and the backwards passes, we can compute the smoothed posterior marginals $P(S_t = k | O_1^T)$. First we find that

$$\begin{aligned} P(S_t = k \cap O_1^T) &= P(O_{t+1}^T \cap S_t = k \cap O_1^t) \\ &= P(O_{t+1}^T | S_t = k) P(S_t = k \cap O_1^t) = \beta_k^{(t)} \alpha_k^{(t)}. \end{aligned}$$

Then we normalise by dividing by $P(O_1^T) = \sum_{k=1}^K \alpha_k^{(t)} \beta_k^{(t)}$ to get

$$P(S_t = k | O_1^T) = \frac{\alpha_k^{(t)} \beta_k^{(t)}}{\sum_{j=1}^K \alpha_j^{(t)} \beta_j^{(t)}}. \tag{30}$$

We can also compute the posterior 'two slice' marginals, $P(S_{t-1} = j \cap S_t = k | O_1^T)$, which describe posterior distributions over *transitions* rather than states. As with the 'one slice' marginals, we compute the joint distribution in terms of things we know already and then normalise to get the conditional distribution.

$$\begin{aligned} P(S_{t-1} = j \cap S_t = k \cap O_1^T) &= P(O_t^T | S_t = k) P(S_t = k | S_{t-1} = j) P(S_{t-1} = j \cap O_1^{t-1}) \\ &= P(O_{t+1}^T | S_t = k) P(O_t^t | S_t = k) a_{jk} \alpha_j^{(t-1)} \\ &= \beta_k^{(t)} B_k^{(t)} a_{jk} \alpha_j^{(t-1)}. \end{aligned}$$

The joint posterior is therefore

$$P(S_{t-1} = j \cap S_t = k | O_1^T) = \frac{\beta_k^{(t)} B_k^{(t)} a_{jk} \alpha_j^{(t-1)}}{\sum_{j'=1}^K \sum_{k'=1}^K \beta_{k'}^{(t)} B_{k'}^{(t)} a_{j'k'} \alpha_{j'}^{(t-1)}}. \tag{31}$$

Both the one slice and the two slice marginals are needed to learn the parameters using the EM algorithm.

5.5 Viterbi algorithm

The Viterbi algorithm solves the decoding problem by providing an efficient way to find the most probable state sequence given the observations, i.e., the sequence $\mathbf{s}$ that maximises the full posterior distribution $p(\mathbf{s}|\mathbf{x})$. Since $p(\mathbf{s}|\mathbf{x}) = p(\mathbf{s},\mathbf{x})/p(\mathbf{x})$, the problem is equivalent to maximising the joint probability:

$$\hat{\mathbf{s}} = \arg \max_{\mathbf{s}\in(1..K)^T} P(S_{1:T} = s_{1:T} \cap X_{1:T} = x_{1:T}) \tag{32}$$

As with the forwards and backwards algorithms, we solve it recursively, in this case, by investigating the following quantity:

$$\delta_k^{(t)} \overset{\text{def}}{=} \max_{\mathbf{s}\in(1..K)^{T-1}} P(S_{1:t-1} = \mathbf{s} \cap S_t = k \cap O_1^t), \tag{33}$$

which is the probability of the most probable event that involves a sequence of states ending in k at time t *and* generates all the observations up till time t. We consider how to compute $\delta_k^{(t)}$ in terms of the $\delta_j^{(t-1)}$ at the previous time step—once again, factorisation of a joint probability is the key step:

$$\begin{aligned}
\delta_k^{(t)} &= \max_{\mathbf{s}\in(1..K)^{t-1}} P(S_{1:t-1} = \mathbf{s} \cap S_t = k \cap O_1^t) \\
&= \max_{\substack{\mathbf{s}\in(1..K)^{t-2} \\ j\in 1..K}} P(S_{1:t-2} = \mathbf{s} \cap S_{t-1} = j \cap S_t = k \cap O_1^{t-1} \cap O_t^t) \\
&= \max_{\substack{\mathbf{s}\in(1..K)^{t-2} \\ j\in 1..K}} P(O_t^t | S_t = k) P(S_t = k | S_{t-1} = j) P(S_{1:t-2} = \mathbf{s} \cap S_{t-1} = j \cap O_1^{t-1}).
\end{aligned}$$

Notice how we have split the maximisation over a sequence of length $t-1$ into a joint maximisation over two variables: an intial subsequence of length $t-2$, and the final element j. Notice also that the dependence on the initial segment $\mathbf{s}$ is entirely contained within the last factor. Hence we can move that part of the maximisation in to the right to obtain, in terms of quantities already defined,

$$\delta_k^{(t)} = \max_{j\in 1..K} B_k^{(t)} a_{jk} \left[\max_{\mathbf{s}\in(1..K)^{t-2}} P(S_{1:t-2} = \mathbf{s} \cap S_{t-1} = j \cap O_1^{t-1}) \right].$$

The final steps are to notice the expression in brackets is the definition of $\delta_j^{(t-1)}$ and that $B_k^{(t)}$ is independent of j, and so we obtain the recurrence relation

$$\delta_k^{(t)} = B_k^{(t)} \max_{j\in 1..K} a_{jk} \delta_j^{(t-1)}, \qquad \delta_k^{(1)} = B_k^{(t)} \pi_k. \tag{34}$$

Once we have computed $\delta_k^{(t)}$ for all k and t up to T, we can work backwards from the end of the sequence to construct the most probable state sequence. If $\delta_k^{(T)}$ is the probability of the best complete sequence that ends in k, then the best sequence overall must end in whatever value of k maximises $\delta_k^{(T)}$, i.e.

$$\hat{s}_T = \arg \max_{k\in 1..K} \delta_k^{(T)}. \tag{35}$$

Having established that the best sequence ends in $\hat{s}_T$, we look back one step to determine $\hat{s}_{T-1}$. Suppose we know that the most probable sequence overall has $\hat{s}_t$ at time t. We already know that $\delta_j^{(t-1)}$ is the probability of the best sequence up to $t-1$ that ends with $S_{t-1}=j$. The probability of the best such sequence followed by a transition to $\hat{s}_t$ is $a_{j,\hat{s}_t}\delta_j^{(t-1)}$. Therefore, if we choose

$$\hat{s}_{t-1} = \arg\max_{j\in 1..K} a_{j,\hat{s}_t}\delta_j^{(t-1)}, \tag{36}$$

then we know that, of all the sequences that arrive at $\hat{s}_t$ at time t, the most probable one must arrive via $\hat{s}_{t-1}$ at time $t-1$. We apply this recursively starting from $\hat{s}_T$, working backwards until we have the whole sequence.

If this argument is not convincing enough, then the following more rigorous derivation may help. Suppose we have already established that the most probable sequence from $t+1$ onwards is $\hat{s}_{t+1:T}$, and we want to find the initial sequence $\hat{s}_{1:t}$ that maximises the probability of the entire resulting sequence $\hat{s}_{1:T}$. We can write this problem as

$$\begin{aligned}\hat{s}_{1:t} &= \arg\max_{\mathbf{s}\in(1..K)^t} P(S_{1:t}=\mathbf{s}\cap S_{t+1:T}=\hat{s}_{t+1:T}\cap O_1^T)\\ &= \arg\max_{\mathbf{s}\in(1..K)^t} P(O_{t+1}^T|S_{t+1:T}=\hat{s}_{t+1:T})P(S_{t+1}=\hat{s}_{t+1}|S_t=s_t)P(S_{1:t}=\mathbf{s}\cap O_1^t).\end{aligned}$$

Note that the first factor is independent of $\mathbf{s}$ and so plays no part in the maximisation and can be ignored. As before, we can split $\hat{s}_{1:t}$ into the initial subsequence $\hat{s}_{1:t-1}$ and the last element $\hat{s}_t$ to obtain

$$(\hat{s}_{1:t-1},\hat{s}_t) = \arg\max_{\substack{\mathbf{s}\in(1..K)^{t-1}\\ j\in 1..K}} a_{j,\hat{s}_{t+1}}P(S_{1:t-1}=\mathbf{s}\cap S_t=j\cap O_1^t),$$

where we have also written the transition probability in terms of the transition matrix. Now suppose that we are only interested in the value of $\hat{s}_t$. Since the dependence on $\mathbf{s}$ is entirely contained in the last factor, we can push the maximisation over $\mathbf{s}$ to the right and retain only the maximisation over j at the outer level:

$$\begin{aligned}\hat{s}_t &= \arg\max_{j\in 1..K} a_{j,\hat{s}_{t+1}}\left[\max_{\mathbf{s}\in(1..K)^{t-1}} P(S_{1:t-1}=\mathbf{s}\cap S_t=j\cap O_1^t)\right]\\ &= \arg\max_{j\in 1..K} a_{j,\hat{s}_{t+1}}\delta_j^{(t)},\end{aligned}$$

which agrees with the expression we had before (36). Note that the maximisation involved here is essentially the same as the one required to computed the $\delta_k^{(t)}$ in the forward pass (34), except that here we need the maximising argument j rather than the maximal value itself. In implementations, it is common to compute both during the forwards pass and store the values

$$\psi_k^{(t)} = \arg\max_{j\in 1..K} a_{jk}\delta_j^{(t-1)}. \tag{37}$$

Then, when the forward pass is complete, we can construct the most probable sequence backwards starting from the end using

$$\hat{s}_{t-1} = \psi_{\hat{s}_t}^{(t)}. \tag{38}$$

6 Learning

In this section we'll apply the EM algorithm to learning the parameters of an HMM. The observed data $\mathbf{x} \in \mathcal{X}^T$ will consist of a sequence $(x_1, \ldots, x_T)$ sampled from the HMM variables $(X_1, \ldots, X_T)$. The latent variables $\mathbf{s}$ correspond to the Markov chain $(S_1, \ldots, S_T)$. Hence, the objective is to optimise the function

$$\mathcal{L}^*(\theta, q) = \sum_{\mathbf{s} \in (1..K)^T} q(\mathbf{s}) \log \frac{p(\mathbf{x}, \mathbf{s}; \theta)}{q(\mathbf{s})}. \tag{39}$$

For a discrete output HMM with $\mathcal{X} = 1..N$, the parameters are $\theta = (\pi, a, b)$, where $\pi \in \mathbb{R}^K$, $a \in \mathbb{R}^{K \times K}$, and $b \in \mathbb{R}^{K \times N}$. These parameters are subject to the normalisation constraints: $\sum_{k=1}^K \pi_k = 1$, and, for all $j \in 1..K$, $\sum_{k=1}^K a_{jk} = 1$ and $\sum_{i=1}^N b_{ji} = 1$. The complete data log likelihood in the (π, a, b) parameterisation is

$$\log p(\mathbf{x}, \mathbf{s}; \pi, a, b) = \log \pi_{s_1} + \sum_{t=1}^{T-1} \log a_{s_t, s_{t+1}} + \sum_{t=1}^{T} \log b_{s_t, x_t}. \tag{40}$$

For a continuous output HMM where all the state-conditional densities are in the same family defined on a parameter space Φ, the parameters are $\theta = (\pi, a, \phi)$, where $\phi \in \Phi^K$ and ϕ_k contains the parameters of the k^{th} state-conditional density, as described in § 4. In the (π, a, ϕ) parameterisation, the log likelihood is

$$\log p(\mathbf{x}, \mathbf{s}; \pi, a, \phi) = \log \pi_{s_1} + \sum_{t=1}^{T-1} \log a_{s_t, s_{t+1}} + \sum_{t=1}^{T} \log f(x_t; \phi_{s_t}). \tag{41}$$

6.1 E step

The E step consists of setting q equal to the posterior distribution using the current parameter estimates, i.e., setting $q(\mathbf{s}) = p(\mathbf{s}|\mathbf{x}; \theta)$. This is potentially quite a complex object since there are K^T possible state sequences. Fortunately, we never need to represent it explicitly; however, we will find it useful to define the following statistics, computed using the various inference algorithms described in § 5, which are needed to implement the M step. Assuming that q is indeed the posterior distribution using the current parameters, the 'one slice' marginals are

$$\gamma_k^{(t)} \stackrel{\text{def}}{=} \sum_{\mathbf{s}} q(\mathbf{s}) \delta_{k, s_t} = P_\theta(S_t = k | O_1^T), \tag{42}$$

where P_θ denotes the probability measure implied by the current parameters θ. These values can be computed, as described in § 5.4, by running the forwards-backwards algorithm with current parameters and using (30). Similarly, the 'two slice' marginals are

$$\xi_{jk}^{(t)} \stackrel{\text{def}}{=} \sum_{\mathbf{s}} q(\mathbf{s}) \delta_{j, s_t} \delta_{k, s_{t+1}} = P_\theta(S_t = j \cap S_{t+1} = k | O_1^T), \tag{43}$$

and can also be computed using the forwards-backwards algorithm (31).

6.2 M step

To implement the M step we introduce a Lagrangian objective function with Lagrange multipliers to manage the constraints on some of the parameters. Depending on which parameterisation we are using, we need to find the stationary points of

$$\mathcal{L}^\lambda(\pi, a, b) = \mathcal{L}^*((\pi, a, b), q) - \lambda^\pi \sum_{j=1}^{K} \pi_j - \sum_{j=1}^{K} \left[\lambda_j^a \sum_{k=1}^{K} a_{jk} + \lambda_j^b \sum_{i=1}^{N} b_{ji} \right] \tag{44}$$

or, if we are using the (π, a, ϕ) parameterisation, of

$$\mathcal{L}^\lambda(\pi, a, \phi) = \mathcal{L}^*((\pi, a, \phi), q) - \lambda^\pi \sum_{j}^{K} \pi_j - \sum_{j=1}^{K} \lambda_j^a \sum_{k=1}^{K} a_{jk} \tag{45}$$

Optimisation of π First, we find that

$$\begin{aligned}\frac{\partial \mathcal{L}^*(\theta, q)}{\partial \pi_k} &= \sum_{\mathbf{s}} q(\mathbf{s}) \frac{\partial}{\partial \pi_k} \log \pi_{s_1} = \sum_{\mathbf{s}} q(\mathbf{s}) \frac{\partial \pi_{s_1}}{\partial \pi_k} \frac{1}{\pi_{s_1}} \\ &= \sum_{\mathbf{s}} q(\mathbf{s}) \delta_{k,s_1} \frac{1}{\pi_{s_1}} \qquad = \left(\sum_{\mathbf{s}} q(\mathbf{s}) \delta_{k,s_1} \right) \frac{1}{\pi_k} = \gamma_k^{(1)} \frac{1}{\pi_k}.\end{aligned}$$

Then we differentiate the constraint:

$$\frac{\partial}{\partial \pi_k} \lambda^\pi \sum_j \pi_j = \lambda^\pi \sum_j \frac{\partial \pi_j}{\partial \pi_k} = \lambda^\pi \sum_j \delta_{jk} = \lambda^\pi$$

The constrained optimum π^* is found by setting these two equal and applying the constraint to solve for both λ^π and π_k:

$$\gamma_k^{(1)} \frac{1}{\pi_k^*} = \lambda^\pi \quad \Longrightarrow \quad \pi_k^* = \frac{\gamma_k^{(1)}}{\lambda^\pi} \quad \Longrightarrow \quad \lambda^\pi = \sum_{j=1}^{K} \gamma_k^{(1)} = 1,$$

where the last step uses the fact that $\gamma_k^{(1)}$ is already a properly normalised posterior distribution. The end result is

$$\pi_k^* = \gamma_k^{(1)}. \tag{46}$$

Optimisation of a The derivatives of the objective with respect to a are

$$\begin{aligned}\frac{\partial \mathcal{L}^\lambda(\theta)}{\partial a_{jk}} &= \sum_{\mathbf{s}} q(\mathbf{s}) \sum_{t=1}^{T-1} \frac{\partial}{\partial a_{jk}} \log a_{s_t, s_{t+1}} - \frac{\partial}{\partial a_{jk}} \sum_{j'=1}^{K} \lambda_{j'}^a \sum_{k'=1}^{K} a_{j'k'} \\ &= \sum_{\mathbf{s}} q(\mathbf{s}) \sum_{t=1}^{T-1} \frac{\partial a_{s_t, s_{t+1}}}{\partial a_{jk}} \frac{1}{a_{s_t, s_{t+1}}} - \sum_{j'=1}^{K} \sum_{k'=1}^{K} \frac{\partial a_{j'k'}}{\partial a_{jk}} \lambda_{j'}^a \\ &= \sum_{t=1}^{T-1} \sum_{\mathbf{s}} q(\mathbf{s}) \delta_{j,s_t} \delta_{k,s_{t+1}} \frac{1}{a_{s_t, s_{t+1}}} - \sum_{j'=1}^{K} \sum_{k'=1}^{K} \delta_{jj'} \delta_{kk'} \lambda_{j'}^a \\ &= \sum_{t=1}^{T-1} \left(\sum_{\mathbf{s}} q(\mathbf{s}) \delta_{j,s_t} \delta_{k,s_{t+1}} \right) \frac{1}{a_{jk}} - \lambda_j^a = \sum_{t=1}^{T-1} \xi_{jk}^{(t)} \frac{1}{a_{jk}} - \lambda_j^a.\end{aligned}$$

Setting these derivative to zero and solving for a_{jk}, we get

$$a^*_{jk} = \frac{1}{\lambda^a_j}\sum_{t=1}^{T-1}\xi^{(t)}_{jk}, \quad \text{but} \quad \sum_{k=1}^{K} a^*_{jk} = 1, \quad \text{so} \quad \lambda^a_j = \sum_{k=1}^{K}\sum_{t=1}^{T-1}\xi^{(t)}_{jk}.$$

The end result is therefore

$$a^*_{jk} = \frac{\sum_{t=1}^{T-1}\xi^{(t)}_{jk}}{\sum_{k=1}^{K}\sum_{t=1}^{T-1}\xi^{(t)}_{jk}}. \tag{47}$$

Optimistion of b Using the same kind of methods as above, we get

$$\begin{aligned}\frac{\partial\mathcal{L}^\lambda(\theta)}{\partial b_{ji}} &= \sum_{\mathbf{s}} q(\mathbf{s})\sum_{t=1}^{T}\frac{\partial}{\partial b_{ji}}\log b_{s_t,x_t} - \frac{\partial}{\partial b_{ji}}\sum_{j',i'}\lambda^b_{j'}b_{j'i'}\\ &= \sum_{t=1}^{T}\left(\sum_{\mathbf{s}} q(\mathbf{s})\delta_{j,s_t}\right)\delta_{i,x_t}\frac{1}{b_{ji}} - \lambda^b_j\\ &= \left(\sum_{t=1}^{T}\gamma^{(t)}_j\delta_{i,x_t}\right)\frac{1}{b_{ji}} - \lambda^b_j\end{aligned}$$

Setting this equal to zero give us

$$b^*_{ji} = \frac{1}{\lambda^b_j}\sum_{t=1}^{T}\gamma^{(t)}_j\delta_{i,x_t}, \quad \text{but} \quad \sum_{i=1}^{K} b^*_{ji} = 1, \quad \text{so} \quad \lambda^b_j = \sum_{i=1}^{K}\sum_{t=1}^{T}\gamma^{(t)}_j\delta_{i,x_t}$$

and therefore

$$b^*_{ji} = \frac{\sum_{t=1}^{T}\gamma^{(t)}_j\delta_{i,x_t}}{\sum_{i'=1}^{K}\sum_{t=1}^{T}\gamma^{(t)}_j\delta_{i',x_t}}. \tag{48}$$

Optimisation of ϕ The derivative of $\mathcal{L}^*(\theta, q)$ with respect to ϕ_k is

$$\begin{aligned}\frac{\partial\mathcal{L}^*(\theta,q)}{\partial\phi_k} &= \sum_{\mathbf{s}} q(\mathbf{s})\sum_{t=1}^{T}\frac{\partial}{\partial\phi_k}\log f(x_t;\phi_{s_t})\\ &= \sum_{t=1}^{T}\gamma^{(t)}_k\frac{\partial}{\partial\phi_k}\log f(x_t;\phi_k).\end{aligned}$$

Zeroing this is equivalent to solving a weighted maximum likelihood problem using the parameterised density function f used for the state-conditional densities. Given the associated weighted maximum likelihood solver $\mathrm{WML}_f : \mathbb{R}^T \times \mathcal{X}^T \to \Phi$, the solution is

$$\phi^*_k = \mathrm{WML}_f(\gamma^{(:)}_k, \mathbf{x}), \tag{49}$$

where $\gamma^{(:)}_k$ denotes the array of T elements formed by taking the values of $\gamma^{(t)}_k$ for all values of t but a given fixed k.

6.3 Summary

One complete iteration of the EM algorithm for HMMs, (aka the Baum Welch algorithm) is a mapping from $(\pi, a, b) \mapsto (\pi^*, a^*, b^*)$ or $(\pi, a, \phi) \mapsto (\pi^*, a^*, \phi^*)$, depending on which parameterisation is being used.

1. Compute the observation likelihoods

$$B_k^{(t)} = f(x_t; \phi_k) \quad \text{or} \quad B_k^{(t)} = b_{k,x_t}.$$

2. Run the forwards-backwards algorithm using the current parameters (π, a) and the observation likelihoods B.

3. Compute the one and two slice marginals:

$$\gamma_k^{(t)} = \frac{\alpha_k^{(t)} \beta_k^{(t)}}{\sum_{j=1}^K \alpha_j^{(t)} \beta_j^{(t)}}, \qquad \xi_{jk}^{(t)} = \frac{\beta_k^{(t)} B_k^{(t)} a_{jk} \alpha_j^{(t-1)}}{\sum_{j'=1}^K \sum_{k'=1}^K \beta_{k'}^{(t)} B_{k'}^{(t)} a_{j'k'} \alpha_{j'}^{(t-1)}}$$

4. Compute the new values of the parameters:

$$\begin{aligned} a_{jk}^* &= \frac{\sum_{t=1}^{T-1} \xi_{jk}^{(t)}}{\sum_{k=1}^K \sum_{t=1}^{T-1} \xi_{jk}^{(t)}}, \qquad \pi_k^* = \gamma_k^{(1)}, \\ \phi_k^* &= \mathrm{WML}_f(\gamma_k^{(:)}, \mathbf{x}) \\ \text{or} \qquad b_{ji}^* &= \frac{\sum_{t=1}^T \gamma_j^{(t)} \delta_{i,x_t}}{\sum_{i'=1}^K \sum_{t=1}^T \gamma_j^{(t)} \delta_{i',x_t}} \end{aligned}$$

In practical implementations, steps 2 and 3 are usually combined, since the one and two slice marginals can be computed relatively cheaply using information that is available inside the implementation of the forwards-backwards algorithm. In addition, the forwards-backwards algorithm is susceptible to numerical underflow and so would not be implemented exactly as specified in § 5.