Lecture topics:

- Markov chains (cont'd)
- Hidden Markov Models

Markov chains (cont'd)

In the context of spectral clustering (last lecture) we discussed a random walk over the nodes induced by a weighted graph. Let $W_{ij} \geq 0$ be symmetric weights associated with the edges in the graph; $W_{ij} = 0$ whenever edge doesn't exist. We also assumed that $W_{ii} = 0$ for all i. The graph defines a random walk where the probability of transitioning from node (state) i to node j is given by

$$P(X(t+1) = j | X(t) = i) = P_{ij} = \frac{W_{ij}}{\sum_{j'} W_{ij'}} \tag{1}$$

Note that self-transitions (going from i to i) are disallowed because $W_{ii} = 0$ for all i. We can understand the random work as a *homogeneous Markov chain*: the probability of transitioning from i to j only depends on i, not the path that took the process to i. In other words, the current state summarizes the past as far as the future transitions are concerned. This is a Markov (conditional independence) property:

$$P(X(t+1) = j | X(t) = i, X(t-1) = i_{t-1}, \ldots, X(1) = i_1) = P(X(t+1) = j | X(t) = i) \tag{2}$$

The term "homogeneous" specifies that the transition probabilities are independent of time (the same probabilities are used whenever the random walk returns to i).

We also defined *ergodicity* as follows: Markov chain is ergodic if there exist a finite m such that

$$P(X(t+m) = j | X(t) = i) > 0 \text{ for all } i \text{ and } j \tag{3}$$

Simple weighted graphs need not define ergodic chains. Consider, for example, a weighted graph between two nodes $1 - 2$ where $W_{12} > 0$. The resulting random walk is necessarily periodic, i.e., $121212\ldots$. A Markov chain is ergodic only when all the states are communicating and the chain is *aperiodic* which is clearly not the case here. Similarly, even a graph $1 - 2 - 3$ with positive weights on the edges would not define an ergodic Markov chain. Every other state would necessarily be 2, thus the chain is periodic. The reason here is

that $W_{ii} = 0$. By adding a positive self-transition, we can remove periodicity (random walk would stay in the same state a variable number of steps). Any connected weighted graph with positive weights and positive self-transitions gives rise to an ergodic Markov chain.

Our definition of the random walk so far is a bit incomplete. We did not specify how the process started, i.e., we didn't specify the initial state distribution. Let $q(i)$ be the probability that the random walk is started from state i. We will use q as a vector of probabilities across k states (reserving n for the number training examples as usual).

There are two ways of describing Markov chains: through *state transition diagrams* or as simple *graphical models*. The descriptions are complementary. A transition diagram is a directed graph over the possible states where the arcs between states specify all allowed transitions (those occuring with non-zero probability). See Figure 1 for examples. We could also add the initial state distribution as transitions from a dedicated initial (null) state (not shown in the figure).

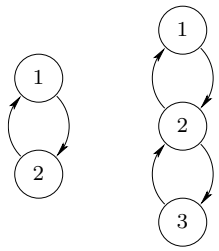

Figure 1: Examples of transition diagrams defining non-ergodic Markov chains.

In graphical models, on the other hand, we focus on explicating variables and their dependencies. At each time point the random walk is in a particular state $X(t)$. This is a random variable. It's value is only affected by the random variable $X(t-1)$ specifying the state of the random walk at the previous time point. Graphically, we can therefore write a sequence of random variables where arcs specify how the values of the variables are influenced by others (dependent on others). More precisely, $X(t-1) \rightarrow X(t)$ means that the value of $X(t)$ depends on $X(t-1)$. Put another way, in simulating the random walk, we would have to know the value of $X(t-1)$ in order to sample a value for $X(t)$. The graphical model is shown in Figure 2.

State prediction

We will cast the problem of calculating the predictive probabilities over states in a form

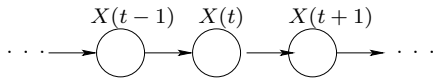

Figure 2: Markov chain as a graphical model.

that will be useful for Hidden Markov Models later on. Since

$$P(X(t+m) = j | X(t) = i) = [P^m]_{ij} \tag{4}$$

we can also write for any n

$$P(X(n) = j) = \sum_{i=1}^{k} q(i) P(X(n) = j | X(1) = i) = \sum_{i=1}^{k} q(i) [P^{n-1}]_{ij} \tag{5}$$

In a vector form $q^T P^{n-1}$ is a row vector whose j^{th} component is $P(X(n) = j)$. Note that the matrix products involve summing over all the intermediate states until $X(n) = j$. More explicitly, let's evaluate the sum over all the states $x_1, \ldots, x_n$ in the matrix form as

$$\sum_{x_1,\ldots,x_n} P(X(1) = x_1) \prod_{t=1}^{n-1} P(X(t+1) = x_{t+1} | X(t) = x_t) = q^T \overbrace{P\, P \cdots P\, P}^{n-1 \text{ times}} \mathbf{1} = 1 \tag{6}$$

This is a sum over k^n possible state configurations (settings of $x_1, \ldots, x_n$) but can be easily performed in terms of matrix products. We can understand this in terms of recursive evaluation of t step probabilities $\alpha_t(i) = P(X(t) = i)$. We will write α_t for the corresponding column vector so that

$$q^T \overbrace{P\, P \cdots P\, P}^{t-1 \text{ times}} = \alpha_t^T \tag{7}$$

Clearly,

$$q^T = \alpha_1^T \tag{8}$$

$$\alpha_{t-1}^T P = \alpha_t^T, \quad t > 1 \tag{9}$$

$$\sum_{i=1}^{k} \alpha_{t-1}(i) P_{ij} = \alpha_t(j) \tag{10}$$

Estimation

Markov models can be estimated easily from observed sequences of states. Given $x_1, \ldots, x_n$ (e.g., 1212221), the log-likelihood of the observed sequence is given by

$$\log P(x_1, \ldots, x_n) = \log \left[P(X(1) = x_1) \prod_{t=1}^{n-1} P(X(t+1) = x_{t+1} | X(t) = x_t) \right] \quad (11)$$

$$= \log q(x_1) + \sum_{t=1}^{n-1} \log P_{x_t, x_{t+1}} \quad (12)$$

$$= \log q(x_1) + \sum_{i,j} \hat{n}(i,j) \log P_{ij} \quad (13)$$

where $\hat{n}(i,j)$ is the number of observed transitions from i to j in the sequence $x_1, \ldots, x_n$. The resulting maximum likelihood setting of P_{ij} is obtained as an empirical fraction

$$\hat{P}_{ij} = \frac{\hat{n}(i,j)}{\sum_{j'} \hat{n}(i,j')} \quad (14)$$

Note that $q(i)$ can only be reliably estimated from multiple observed sequences. For example, based on $x_1, \ldots, x_n$, we would simply set $\hat{q}(i) = \delta(i, x_1)$ which is hardly accurate (sample size one). Regularization is useful here, as before.

Hidden Markov Models

Hidden Markov Models (HMMs) extend Markov models by assuming that the states of the Markov chain are not observed directly, i.e., the Markov chain itself remains hidden. We therefore also model how the states relate to the actual observations. This assumption of a simple Markov model underlying a sequence of observations is very useful in many practical contexts and has made HMMs very popular models of sequence data, from speech recognition to bio-sequences. For example, to a first approximation, we may view words in speech as Markov sequences of phonemes. Phonemes are not observed directly, however, but have to be related to acoustic observations. Similarly, in modeling protein sequences (sequences of amino acid residues), we may, again approximately, describe a protein molecule as a Markov sequence of structural characteristics. The structural features are typically not observable, only the actual residues.

We can understand HMMs by combining mixture models and Markov models. Consider the simple example in Figure 3 over four discrete time points $t = 1, 2, 3, 4$. The figure summarizes multiple sequences of observations $y_1, \ldots, y_4$, where each observation sequence

corresponds to a single value y_t per time point. Let's begin by ignoring the time information and instead collapse the observations across the time points. The observations form two clusters are now well modeled by a two component mixture:

$$P(y) = \sum_{j=1}^{2} P(j)P(y|j) \tag{15}$$

where, e.g., $P(y|j)$ could be a Gaussian $N(y; \mu_j, \sigma_j^2)$. By collapsing the observations we are effectively modeling the data at each time point with the same mixture model. If we generate data form the resulting mixture model we would select a mixture component at random at each time step and generate the observation from the corresponding component (cluster). There's nothing that ties the selection of mixture components in time so that samples from the mixture yield "phantom" clusters at successive time points (we select the wrong component/cluster with equal probability). By omitting the time information, we therefore place half of the probability mass in locations with no data. Figure 4 illustrates the mixture model as a graphical model.

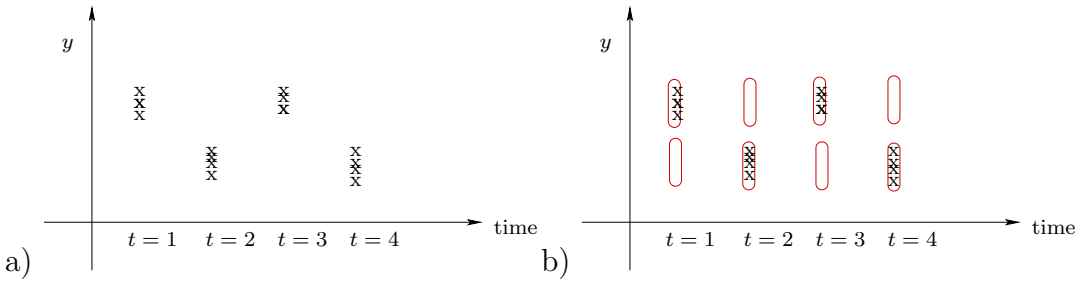

Figure 3: a) Example data over four time points, b) actual data and ranges of samples generated from a mixture model (red ovals) estimated without time information.

The solution is to model the selection of the mixture components as a Markov model, i.e., the component at $t = 2$ is selected on the basis of the component used at $t = 1$. Put another way, each state in the Markov model now uses one of the components in the mixture model to generate the corresponding observation. As a graphical model, the mixture model is a combination of the two as shown in Figure 5.

Probability model

One advantage of representing the HMM as a graphical model is that we can easily write down the joint probability distribution over all the variables. The graph explicates how the

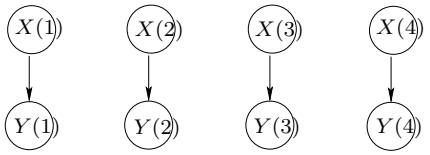

Figure 4: A graphical model view of the mixture model over the four time points. The variables are indexed by time (different samples would be drawn at each time point) but the parameters are shared across the four time points. $X(t)$ refers to the selection of the mixture component while $Y(t)$ refers to the observations.

variables depend on each other (who influences who) and thus highlights which conditional probabilities we need to write down:

$$P(x_1, \ldots, x_n, y_1, \ldots, y_n) = P(x_1)P(y_1|x_1)P(x_2|x_1)P(y_2|x_2)\ldots \quad (16)$$

$$= P(x_1)P(y_1|x_1)\prod_{t=1}^{n-1}[P(x_{t+1}|x_t)P(y_{t+1}|x_{t+1})] \quad (17)$$

$$= q(x_1)P(y_1|x_1)\prod_{t=1}^{n-1}\left[P_{x_t,x_{t+1}}P(y_{t+1}|x_{t+1})\right] \quad (18)$$

where we have used the same notation as before for the Markov chains.

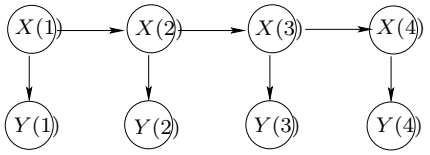

Figure 5: HMM as a graphical model. It is a Markov model where each state is associated with a distribution over observations. Alternatively, we can view it as a mixture model where the mixture components are selected in a time dependent manner.

Three problems to solve

We typically have to be able to solve the following three problems in order to use these models effectively:

1. Evaluate the probability of observed data or

$$P(y_1, \ldots, y_n) = \sum_{x_1,\ldots,x_n} P(x_1, \ldots, x_n, y_1, \ldots, y_n) \tag{19}$$

2. Find the most likely hidden state sequence $x_1^*, \ldots, x_n^*$ given observations $y_1, \ldots, y_n$, i.e.,

$$\{x_1^*, \ldots, x_n^*\} = \arg \max_{x_1,\ldots,x_n} P(x_1, \ldots, x_n, y_1, \ldots, y_n) \tag{20}$$

3. Estimate the parameters of the model from multiple sequences of $y_1^{(l)}, \ldots, y_{n_l}^{(l)}$, $l = 1, \ldots, L$.

Problem 1

As in the context of Markov chains we can efficiently sum over the possible hidden state sequences. Here the summation means evaluating $P(y_1, \ldots, y_n)$. We will perform this in two ways depending on whether the recursion moves forward in time, computing $\alpha_t(j)$, or backward in time, evaluating $\beta_t(i)$. The only change from before is the fact that whatever state we happen to visit at time t, we will also have to generate the observation y_t from that state. This additional requirement of generating the observations can be included via diagonal matrices

$$D_y = \begin{bmatrix} P(y|1) & & 0 \\ & \cdots & \\ 0 & & P(y|k) \end{bmatrix} \tag{21}$$

So, for example,

$$q^T D_{y_1} \mathbf{1} = \sum_{i=1}^{k} q(i) P(y_1|i) = P(y_1) \tag{22}$$

Similarly,

$$q^T D_{y_1} P D_{y_2} \mathbf{1} = \sum_{i=1}^{k} q(i) P(y_1|i) \sum_{j=1}^{k} P_{ij} P(y_2|j) = P(y_1, y_2) \tag{23}$$

We can therefore write the forward and backward algorithms as methods that perform the matrix multiplications in

$$q^T D_{y_1} P D_{y_2} P \cdots P D_{y_n} \mathbf{1} = P(y_1, \ldots, y_n) \tag{24}$$

either in the forward or backward direction. In terms of the forward pass algorithm:

$$q^T D_{y_1} = \alpha_1^T \tag{25}$$

$$\alpha_{t-1}^T P D_{y_t} = \alpha_t^T, \text{ or equivalently} \tag{26}$$

$$\left(\sum_{i=1}^{k} \alpha_{t-1}(i) P_{ij} \right) P(y_t|j) = \alpha_t(j) \tag{27}$$

These values hold exactly $\alpha_t(j) = P(y_1, \ldots, y_t, X(t) = j)$ since we have generated all the observations up to and including y_t and have summed over all the states except for the last one $X(t)$.

The backward pass algorithm is similarly defined as:

$$\beta_n = \mathbf{1} \tag{28}$$

$$\beta_t = P D_{y_{t+1}} \beta_{t+1}, \text{ or equivalently} \tag{29}$$

$$\beta_t(i) = \sum_{j=1}^{k} P_{ij} P(y_{t+1}|j) \beta_{t+1}(j) \tag{30}$$

In this case $\beta_t(i) = P(y_{t+1}, \ldots, y_n | X(t) = i)$ since we have summed over all the possible values of the state variables $X(t+1), \ldots, X(n)$, starting from a fixed $X(t) = i$, and the first observation we have generated in the recursion is y_{t+1}.

By combining the two recursions we can finally evaluate

$$P(y_1, \ldots, y_n) = \alpha_t^T \beta_t = \sum_{i=1}^{k} \alpha_t(i) \beta_t(i) \tag{31}$$

which holds for any $t = 1, \ldots, n$. You can understand this result in two ways: either in terms of performing the remaining matrix multiplication corresponding to the two parts

$$P(y_1, \ldots, y_n) = \overbrace{(q^T D_{y_1} P \cdots P D_{y_t})}^{\alpha_t^T} \overbrace{(P D_{y_{t+1}} \cdots P D_{y_n} \mathbf{1})}^{\beta_t} \tag{32}$$

or as an illustration of the Markov property:

$$P(y_1, \ldots, y_n) = \sum_{i=1}^{k} \overbrace{P(y_1, \ldots, y_t, X(t) = i)}^{\alpha_t(i)} \overbrace{P(y_{t+1}, \ldots, y_n | X(t) = i)}^{\beta_t(i)} \tag{33}$$

Also, since $\beta_n(i) = 1$ for all i, clearly

$$P(y_1, \ldots, y_n) = \sum_{i=1}^{k} \alpha_n(i) = \sum_{i=1}^{k} P(y_1, \ldots, y_t, X(t) = i) \tag{34}$$