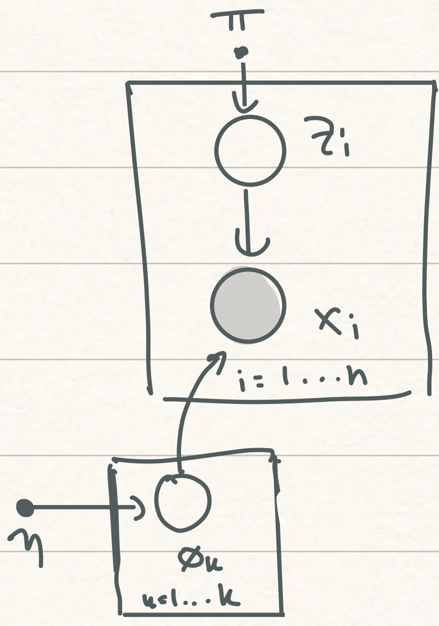
Review on mixture models

Say we have the following generative process:

$$\phi_k \overset{ind}{\sim} f(\eta) \qquad k=1\dots K$$

$$z_i \overset{ind}{\sim} \mathrm{Cat}(\vec{\pi}) \qquad i=1\dots n$$

$$x_i \mid z_i = k \overset{ind.}{\sim} g(\phi_k) \qquad i=1\dots n$$



Now consider the likelihood:

$$p(x_{1:n} \mid \Phi, \pi) = \underbrace{\sum_{z_1} \cdots \sum_{z_n}}_{K^n \text{ terms}} p(z_{1:n}, x_{1:n} \mid \Phi, \pi)$$

Last time we said this involved many summands. This is true, but slightly misleading.
Why? Because conditional on the parameters, the likelihood factorizes, and the sums distribute:

$$= \sum_{z_1} \cdots \sum_{z_n} \prod_{i=1}^{n} p(x_i, z_i \mid \Phi, \pi)$$

$$= \prod_{i=1}^{n} \left[\sum_{z_i} p(x_i, z_i \mid \Phi, \pi) \right] = \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k \, g(x_i; \phi_k)$$

There's nothing intractable about this. This means the complete conditional of Z also factorizes:

$$p(z_{1:n} \mid x_{1:n}, \Phi, \pi) = \prod_{i=1}^{n} p(z_i \mid x_i, \Phi, \pi)$$

However, all of this is conditional on knowing hThe combinatoric intractability of mixture models arises when we do not know the parameters. Consider marginalizing out the likelihood parameters:

$$p(x_{1:n} \mid \eta, \pi) = \sum_{z_1} \cdots \sum_{z_n} \underbrace{\int \prod_{i=1}^{n} p(x_i; z_i \mid \Phi, \pi) \, f(\Phi; \eta) \, d\Phi}_{p(x_{1:n}, z_{1:n} \mid \eta, \pi)}$$

We now get a term that does not factorize over data points, and the sums no longer distribute in.

This is why algorithms like EM are useful: they allow us to fit intractable models in a way that exploits nice conditional structure.

Hidden Markov Models (HMMs)

HMMs are a class of models that are one step more complex than mixture models. In this case, the added complexity will make it so that the evidence is tractable, but the joint posterior $P(Z \mid X)$ is not.

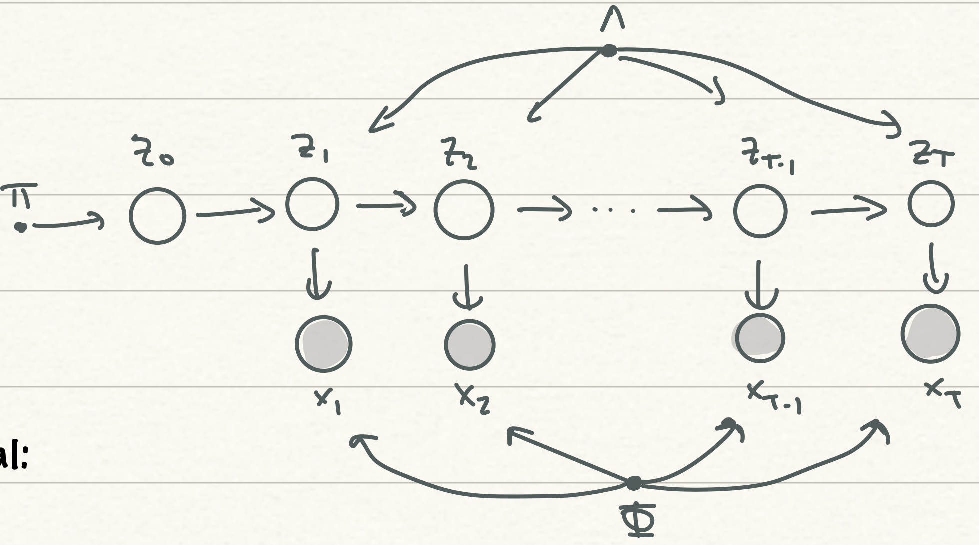
Generative process:

$z_0 \sim p(z_0)$

for $t = 1 \dots T$

$\quad z_t \sim p(z_t \mid z_{t-1})$

$\quad x_t \sim p(x_t \mid z_t)$



"Latent states" are categorical:

$z_t \in \{1 \dots K\}$

"initial distribution"

$p(z_0 = k) = \pi_k, \qquad \sum_k \pi_k = 1$

"transition distribution" **"transition matrix" is a KxK row-stochastic matrix**

$p(z_t = k \mid z_{t-1} = j) = \Lambda_{jk}, \qquad \sum_k \Lambda_{jk} = 1$

"emission distribution" (i.e., class-conditional likelihood) **can be anything:**

$p(x_t \mid z_t = k) = g(x_t \mid \phi_k), \qquad x_t \in \mathbb{R}^D$

Example of applications:

Speech recognition

$x_t \in \mathbb{R}^D$	acoustic signals
$z_t \in$ WORDS	words
$p_\Lambda(z_t \mid z_{t-1})$	language model
$p_\Phi(x_t \mid z_t)$	word-to-speech model

Activity recognition

$x_t \in \mathbb{R}^D$	video frame of subject
$z_t \in \{\text{RUN}, \text{WALK}, \dots\}$	activity of subject
$p_\Lambda(z_t \mid z_{t-1})$	activity-to-activity transition model
$p_\Phi(x_t \mid z_t)$	activity-to-video model

Inference queries of interest:

$p(z_t \mid x_{1:t})$ **"filtering"** $p(z_{t+1} \mid x_{1:t})$ **"predictive"**

$p(z_t \mid x_{1:T})$ **"smoothing"** $p(x_{t+1} \mid x_{1:t})$ **"forecasting"**

$p(x_t \mid x_{1:T \setminus t})$ **"imputing"** $p(x_{1:T})$ **"evidence"**

Learning in HMMs with EM:

$$\underset{\theta}{\operatorname{argmax}}\; p_\theta(x_{1:T}), \qquad \theta = \underbrace{\{\Lambda, \Pi, \Phi\}}_{}$$

all parameters

Evidence lower bound (ELBO):

$$\log p_\theta(x_{1:T}) \geq \mathbb{E}_q\left[\log \frac{p_\theta(x_{1:T}, z_{1:T})}{q(z_{1:T})}\right] \equiv B(q, \theta)$$

E-step: we know from last time, that the optimal E-step update is:

$$\underset{q}{\operatorname{argmax}}\; B(q,\theta) = p(z_{1:T} \mid x_{1:T}, \theta)$$

...however the Z's are no longer conditionally independent in their posterior due to the dependence in their prior. So this posterior has K^T terms and is not tractable to compute. Nevertheless, we will be able to do EM by computing **only the posterior expectations needed** by the M-step. To see which ones are needed, let's look at the M-step...

M-step: we also know from last time that the optimal M-step update is:

$$\underset{\theta}{\operatorname{argmax}}\; B(q,\theta) = \underset{\theta}{\operatorname{argmax}}\; \mathbb{E}_q\left[\log p_\theta(x_{1:T}, z_{1:T})\right]$$

$$= \underset{\theta}{\operatorname{argmax}} \; \mathbb{E}_q \left[\log p_\theta (x_{1:T} \mid z_{1:T}) \right] + \mathbb{E}_q \left[\log p(z_{1:T}) \right]$$

$$= \underset{\phi, \pi, \Lambda}{\operatorname{argmax}} \; \mathbb{E}_q \left[\log \prod_{t=1}^{T} \prod_{k=1}^{K} g(x_t \mid \phi_k)^{1(z_t = k)} \right]$$

$$+ \mathbb{E}_q \left[\log \prod_{t=1}^{T} \prod_{j=1}^{K} \prod_{k=1}^{K} p(z_t = k \mid z_{t-1} = j)^{1(z_t = k, z_{t-1} = j)} \right]$$

$$+ \mathbb{E}_q \left[\log \prod_{k=1}^{K} p(z_0 = k)^{1(z_0 = k)} \right]$$

So this splits into three separate optimization problems:

$$\underset{\phi}{\operatorname{argmax}} \; \sum_t \sum_k \mathbb{E}_q \left[1(z_t = k) \right] \log g(x_t \mid \phi_k)$$

$$\underset{\Lambda}{\operatorname{argmax}} \; \sum_t \sum_j \sum_k \mathbb{E}_q \left[1(z_t = k, z_{t-1} = j) \right] \log \Lambda_{jk}$$

$$\underset{\pi}{\operatorname{argmax}} \; \sum_k \mathbb{E}_q \left[1(z_0 = k) \right] \log \pi_k$$

where the only thing we need from q(...) are the **singleton** and **pairwise marginals.**

E-step: So if we can compute the following "**beliefs**" (i.e., posterior marginals) in the E-step, it will be as if we computed the full posterior (since the **M-step** only needs beliefs):

full posterior (intractable)

$$q(z_{1:T}) = p(z_{1:T} \mid x_{1:T}, \theta)$$

singleton "beliefs"

$$\mathbb{E}_q \left[1(z_t = k) \right] \equiv q(z_t = k) = p(z_t = k \mid x_{1:T}, \theta) \quad \forall k$$

pairwise "beliefs"

$$\mathbb{E}_q \left[1(z_t = k, z_{t-1} = j) \right] = p(z_t = k, z_{t-1} = j \mid x_{1:T}, \theta) \quad \forall j, k$$

Inference in HMMs: the forwards-backwards algorithm

Even though the full posterior over latent states is intractable, it turns out that we can still compute the posterior marginals tractably using a **"dynamic programming" algorithm.**

Forward pass:

Consider the following **predictive marginal:**

$$p(z_{t+1}=k \mid x_{1:t}) = \frac{p(z_{t+1}=k, x_{1:t})}{\sum_j p(z_{t+1}=j, x_{1:t})}$$

Define the numerator to be a function of just z_{t+1} :

$$\alpha_{t+1}(z_{t+1}) \equiv p(z_{t+1}, x_{1:t}) = \underbrace{\sum_{z_t} \cdots \sum_{z_1}}_{k^t \text{ terms}} p(z_{t+1}, z_{1:t}, x_{1:t})$$

Marginalizing over all previous states involves many terms, however the sums distribute

$$= \sum_{z_t} p(z_{t+1} \mid z_t)\, p(x_t \mid z_t) \sum_{z_{t-1}} \cdots \sum_{z_1} p(z_t, z_{1:t-1}, x_{1:t-1})$$

Factoring out only the first term reveals a recursion:

$$= \sum_{z_t} p(z_{t+1} \mid z_t)\, p(x_t \mid z_t)\, \alpha_t(z_t)$$

Going back all the way to t=2:

$$\alpha_2(z_2=k) = p(z_2=k, x_1) = \sum_{j=1}^{k} p(z_2=k \mid z_1=j)\, p(x_1 \mid z_1=j)\, \alpha_1(z_1=j)$$

The "base case" of the recursion is at t=1:

$$\alpha_1(z_1=k) = p(z_1=k) = \sum_{j=1}^{k} p(z_1=k \mid z_0=j)\, p(z_0=j) = \sum_j A_{kj} \pi_j$$

The forward pass in vectorized form:

$$\vec{\alpha}_t = \begin{bmatrix} \alpha_t(1) \\ \vdots \\ \alpha_t(K) \end{bmatrix}, \qquad \vec{l}_t = \begin{bmatrix} l_t(1) \\ \vdots \\ l_t(K) \end{bmatrix} \qquad \text{where} \quad l_t(k) \equiv p(x_t \mid z_t = k)$$

Hadamard (elementwise) product

$$\boxed{\begin{array}{l} \vec{\alpha}_1 = \Lambda^T \pi \\ \text{for } t = 2 \dots T+1: \\ \qquad \vec{\alpha}_t = \Lambda^T (\vec{\alpha}_{t-1} \odot \vec{l}_{t-1}) \end{array}}$$

This involves T matrix-times-vector products that are each O(K^2).

What marginals can we compute with just the forward pass?

$$\alpha_{T+1}(k) = p(z_{T+1} = k, x_{1:T})$$

$$\Downarrow$$

$$p(z_{T+1} = k \mid x_{1:T}) = \frac{\alpha_{T+1}(k)}{\sum_j \alpha_{T+1}(j)}$$

Question: what does the denominator compute?

$$\sum_j \alpha_{T+1}(j) = ? \qquad \qquad \sum_j \alpha_t(j) = ?$$

What marginals can't we compute with only the forward pass?

$$p(z_t \mid x_{1:T}) \quad \text{for} \quad t < T+1$$

These are the beliefs we need for EM, so we'll also need the **backward pass.**

Backward pass:

$$p(z_t = k \mid x_{1:T}) \propto p(z_t = k, x_{1:T})$$

$$p(z_t, x_{1:T}) = \sum_{z_T} \cdots \sum_{z_{t+1}} \sum_{z_{t-1}} \cdots \sum_{z_1} p(z_{1:T}, x_{1:T})$$

$$= \sum_{z_T} \cdots \sum_{z_{t+1}} \sum_{z_{t-1}} \cdots \sum_{z_1} p(z_{t+1:T}, x_{t+1:T} \mid z_t)\, p(x_t \mid z_t)\, p(z_{1:t}, x_{1:t-1})$$

$$= p(x_t \mid z_t) \left[\sum_{z_T} \cdots \sum_{z_{t+1}} p(z_{t+1:T}, x_{t+1:T} \mid z_t) \right] \left[\sum_{z_{t-1}} \cdots \sum_{z_1} p(z_t, z_{1:t-1}, x_{1:t-1}) \right]$$

$$= \underbrace{p(x_t \mid z_t)}_{= \ell_t(z_t)} \; \underbrace{p(x_{t+1:T} \mid z_t)}_{= \beta_t(z_t)} \; \underbrace{p(z_t, x_{1:t-1})}_{= \alpha_t(z_t)}$$

These are the missing pieces: the backward messages.

$$\beta_t(k) = p(x_{t+1:T} \mid z_t = k) = \sum_{z_T} \cdots \sum_{z_{t+1}} p(z_{t+1:T}, x_{t+1:T} \mid z_t = k)$$

Factoring out only the last state reveals another recursion:

$$= \sum_{j=1} \underbrace{p(z_{t+1} = j \mid z_t = k)}_{= \Lambda_{kj}} \underbrace{p(x_{t+1} \mid z_{t+1} = j)}_{= \ell_{t+1}(j)} \underbrace{\sum_{z_{t+2}} \cdots \sum_{z_T} p(z_{t+2:T}, x_{t+2:T} \mid z_{t+1} = j)}_{= p(x_{t+2:T} \mid z_{t+1} = j) \equiv \beta_{t+1}(j)}$$

Going back to T-1:

$$\beta_{T-1}(k) = p(x_T \mid z_{T-1} = k) = \sum_j \Lambda_{kj}\, \ell_T(j)\, \beta_T(j)$$

The "base case" is then at T:

$$\beta_T(k) = 1$$

The backward pass in vectorized form:

$$\vec{\beta}_t = \begin{bmatrix} \beta_t(z_t=1) \\ \vdots \\ \beta_t(z_t=K) \end{bmatrix} : \quad \boxed{\begin{array}{l} \vec{\beta}_T = 1_K \\ \text{for } t = T-1, \dots, 0: \\ \quad \vec{\beta}_t = \Lambda(\vec{l}_{t+1} \odot \vec{\beta}_{t+1}) \end{array}}$$

Computing singleton beliefs:

$$\mathbb{E}_q[\mathbb{1}(z_t=k)] = p(z_t=k \mid x_{1:T}) = \frac{p(z_t=k, x_{1:T})}{\sum_j p(z_t=j, x_{1:T})}$$

$$= \frac{p(z_t=k, x_{1:t-1})\, p(x_t \mid z_t=k)\, p(x_{t+1:T} \mid z_t=k)}{\sum_j \quad \cdots}$$

$$= \frac{\alpha_t(k)\, l_t(k)\, \beta_t(k)}{\sum_j \alpha_t(j)\, l_t(j)\, \beta_t(j)}$$

So we just need to run the forward and backward pass once (each), and we have everything we need to compute the singleton beliefs in O(TK^2).

Computing pairwise beliefs:

$$p(z_t=k, z_{t-1}=j \mid x_{1:T}) = \frac{p(z_t=k, z_{t-1}=j \mid x_{1:T})}{\sum_{k'}\sum_{j'} p(z_t=k', z_{t-1}=j' \mid x_{1:T})}$$

$$p(z_t=k, z_{t-1}=j \mid x_{1:T})$$

$$= p(z_{t-1}=j \mid x_{1:t-2})\, p(x_{t-1} \mid z_{t-1}=j)\, p(z_t=k \mid z_{t-1}=j)\, p(x_t \mid z_t=k)\, p(x_{t+1}:T \mid z_t=k)$$

$$= \alpha_{t-1}(j)\, l_{t-1}(j)\, \Lambda_{jk}\, l_t(k)\, \beta_t(k)$$

Historical notes:

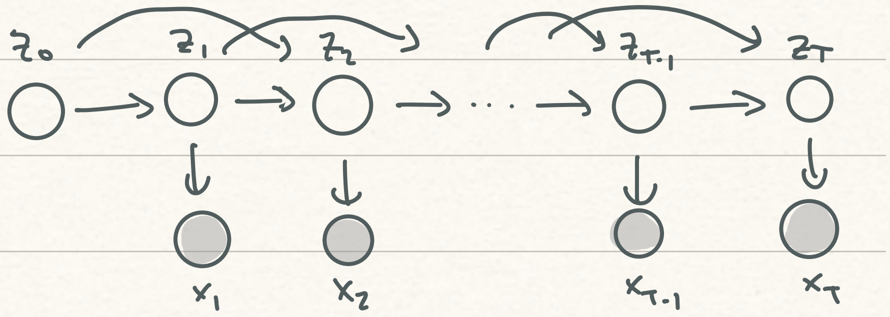
- The forwards-backwards algorithm goes back at least to Rainer & Juang (1986) and is a special case of **belief propagation** and the **sum-product algorithm** (next time).
- EM for HMMs is known as the **Baum-Welch algorithm** and it was invented in 1970 before the general EM algorithm by Dempster et al. (1977)!

M-order HMMs

The latent Markov transition process is now M-order:

$$z_t \sim p(z_t \mid z_{t-1:t-M})$$

e.g. $M=2$



(The **transition matrix** is now MK x K.)

EM would now require the M-order marginals to learn the transition matrix:

$$\mathbb{E}_q[\mathbb{1}(z_t \cdots z_{t+M})] = p(z_t \cdots z_{t+M} \mid x_{1:T}, \theta)$$

These become intractable to compute exactly for larger M.

However, if we could sample sequences of latent states, we could instead compute a Monte Carlo estimate of the belief:

$$\approx \frac{1}{S}\sum_{s=1}^{S} \mathbb{1}(z_t^s = z_t \; \ldots \; z_{t+m}^s = z_{t+m})$$

$$\text{where} \quad z_1^s \cdots z_T^s \overset{iid}{\sim} p(z_1 \cdots z_T \mid x_{1:T})$$

This is called Monte Carlo EM (MC-EM). For S=1 it is called stochastic EM. We only need to be able to draw samples of the latent states from their full posterior.

Forward-filtering backward sampling (FFBS)

① Forward pass : $\alpha_t(u)$

② $z_T \sim p(z_T \mid x_{1:T})$

③ for $t = T-1 \ldots 0$:

$$z_t \sim p(z_t \mid z_{t+1}, x_{1:T})$$

MAP inference in HMMs: the Viterbi algorithm

We won't go through the details today, but another important connection is the Viterbi algorithm, which finds the maximum posterior assignment of latent states in HMMs. The main idea is that the max operation pushes across products (like the sum operation):

$$\max_{z_{1:T}} p(z_{1:T} \mid x_{1:T})$$

$$= \max_{z_1} p(z_1 \mid x_{1:T}) \cdots \max_{z_T} p(z_T \mid z_{T-1}, x_{1:T})$$

This allows us to derive an efficient dynamic programming algorithm. The Viterbi algorithm is a special case of the **max-product algorithm** (next time).

The search for the USS Scorpion

In May 1968 a US Navy submarine (the USS Scorpion) disappeared, carrying two nuclear missiles and 99 crew. There was an urgent search-and-rescue operation led by the USS Mizar on which statistician John Craven pioneered fundamental techniques in Bayesian search search theory to find the missing sub.

The Navy created a search grid of 140 square miles in the Atlantic. Based on measurements (e.g., sonar, magnetometer readings, ...) and searches, they updated a posterior distribution over which cell in the grid the missing sub was.

One important aspect of their search was expert-elicited Markov-chain priors. They asked submarine Captains questions like "if you were in this cell at this time, and in this scenario, what would you do next?". Based on their answers, the statisticians developed a Markov transition model describing how probable the sub would transition from cell j to cell k at time t. In your next homework, you will use your knowledge of HMMs to find the Scorpion.