STAT 5630, Fall 2019

Model Based Clustering and EM Algorithm

Xiwei Tang, Ph.D. <xt4yj@virginia.edu>

University of Virginia
November 5, 2019

Outline

- Model Based Clustering: Gaussian Mixture Models
- The EM Algorithm
- Hidden Markov Models

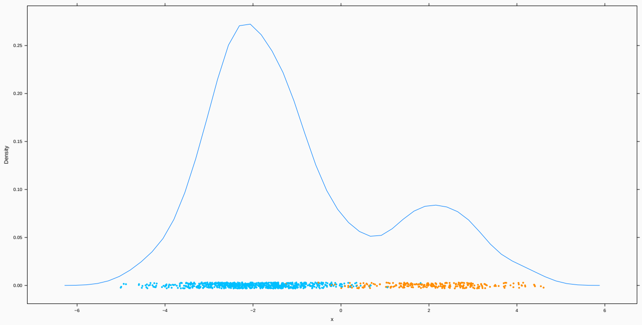
Gaussian Mixture Models

Clustering Retrieve

- K-means and Hierarchical clustering
  - **Hard clustering**
  - Distance based, Nonparametric
- Analogously, for classification:
  - LDA, QDA, Tree, etc. : **Hard classification**
  - Logistic regression: **Soft classification**

Soft Clustering: Gaussian Mixture Model

- Suppose
  - We know that there are two populations, with means μ_1 and μ_2, respectively, and variance $\sigma^2 = 1$.
  - X is the observed outcome (from one of the two populations with probability π)
  - From only the observed data $\{x_i\}_{i=1}^n$, we want to estimate the two population means and the mixing probability: $\boldsymbol{\theta} = (\mu_1, \mu_2, \pi)$.



Gaussian Mixture

- The density of X is a mixture of two Gaussian:

$$\mathsf{p}_X(x) = (1-\pi)\phi_{\mu_1}(x) + \pi\phi_{\mu_2}(x)$$

  where ϕ_μ is the density function of $\mathcal{N}(\mu, 1)$.

- The log-likelihood function based on n observed training data is

$$\ell(\mathbf{x}|\boldsymbol{\theta}) = \sum_{i=1}^{n} \log\left[(1-\pi)\phi_{\mu_1}(x_i) + \pi\phi_{\mu_2}(x_i)\right]$$

- π is the mixing proportion
- $\phi_{\mu_1}(x_i)$ and $\phi_{\mu_2}(x_i)$ are component densities
- Of course, we can solve this by gradient descent, however, that is often slow.

A Different View of GM

- We can look at the GM from another perspective: consider random variable (Z, X)

$$Z \sim Bernoulli(\pi)$$
$$X|Z = 0 \sim N(\mu_1, 1)$$
$$X|Z = 1 \sim N(\mu_2, 1)$$

- Refers to an incomplete data case since the indicator variable $Z \in \{0, 1\}$ is a hidden variable (not observable) that indicates the population label, with $P(Z = 1) = \pi$.
- Hence, we can treat the hidden labels Z as a "missing variables" and use the EM algorithm.

Gaussian Mixture: The EM algorithm

- Instead of directly optimizing $\ell(\mathbf{x}|\boldsymbol{\theta})$, we incorporate the latent variable Z, and write the joint log-likelihood as

$$\begin{aligned}\ell(\mathbf{x},\mathbf{z}|\boldsymbol{\theta}) = &\sum_{i=1}^{n}\left[(1-z_i)\log\phi_{\mu_1}(x_i) + z_i\log\phi_{\mu_2}(x_i)\right] \\ &+ \sum_{i=1}^{n}\left[(1-z_i)\log(1-\pi) + z_i\log\pi\right]\end{aligned}$$

- EM algorithm: We will then optimize this likelihood function by iteratively updating the unknowns: $\mathbf{z}$ and $\boldsymbol{\theta}$.

— At the E-step (expectation), we treat both $\mathbf{x}$ and (μ_1, μ_2, π) as known, and calculate the conditional probability of each z_i.

— At the M-step (maximization), we treat $\mathbf{x}$ and $\mathbf{z}$ as known, and solve the parameters by maximizing the likelihood.

Gaussian Mixture: E-step

- E-step, if both $\mathbf{x}$ and $\boldsymbol{\theta} = (\mu_1, \mu_2, \pi)$ are known, then the conditional probability of each z_i can be calculated as:

$$\begin{aligned} \mathsf{P}(Z_i = 1 | \boldsymbol{\theta}^{(k)}, \mathbf{x}) &= \frac{\mathsf{p}(Z_i = 1, x_i | \boldsymbol{\theta}^{(k)})}{\mathsf{p}(x_i | \boldsymbol{\theta}^{(k)})} \\ &= \frac{\mathsf{p}(Z_i = 1, x_i | \boldsymbol{\theta}^{(k)})}{\mathsf{p}(Z_i = 1, x_i | \boldsymbol{\theta}^{(k)}) + \mathsf{p}(Z_i = 0, x_i | \boldsymbol{\theta}^{(k)})} \end{aligned}$$

- This is pretty simple since we just need to calculate the densities functions of each x_i under the current parameter $\boldsymbol{\theta}^{(k)}$ for each possible label ($z_i = 0$ or 1).

Gaussian Mixture: E-step

- Lets first set up the initial values and estimate the conditional probabilities for each z_i

```
> # generate the data:
> n = 1000; x1 = rnorm(n, mean=-2)
> x2 = rnorm(n, mean=2); z = (runif(n) <= 0.25)
> x = ifelse(z, x2, x1)
>
> # lets setup some (arbitrary) initial values:
> hat_PI = 0.5
> hat_mu1 = -0.25
> hat_mu2 = 0.25
>
> # E step
> # calculate the conditional distribution of the hidden variable z
> d1 = hat_PI * dnorm(x, mean= hat_mu1)
> d2 = (1-hat_PI) * dnorm(x, mean= hat_mu2)
> ez = d2/(d1 + d2)
```

Gaussian Mixture: M-step

- Now we already have $\mathsf{p}(\mathbf{Z} = \mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^{(k)})$, we can replace all the z_i values (since they are unknown anyways) in the likelihood function $\ell(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})$ by their expectations (from the E-step).

$$\widehat{p}_i = \mathsf{p}(Z_i = 1|\mathbf{x}, \boldsymbol{\theta}^{(k)})$$

- After this, things remained in the likelihood only involves $\mathbf{x}$ and $\boldsymbol{\theta}$, so we can solve (the M-step) for the "MLE" of $\boldsymbol{\theta}$ based on this new likelihood function.
- It turns out that these estimators are just weighted means:

$$\widehat{\mu}_1 = \frac{\sum_{i=1}^n (1 - \widehat{p}_i) x_i}{\sum_{i=1}^n (1 - \widehat{p}_i)}, \quad \widehat{\mu}_2 = \frac{\sum_{i=1}^n \widehat{p}_i x_i}{\sum_{i=1}^n \widehat{p}_i} \quad \text{and} \quad \widehat{\pi} = \sum_{i=1}^n \widehat{p}_i / n$$

- We will then iterate the E- and M- steps until convergence.

The EM algorithm: M-step

```
> # M-step
> # based on the conditional distribution, calculate the new MLE
    of the parameters
> PI = mean(ez)
> hat_mu1 = sum( (1-ez) * x ) / sum(1-ez)
> hat_mu2 = sum( ez * x ) / sum(ez)
```

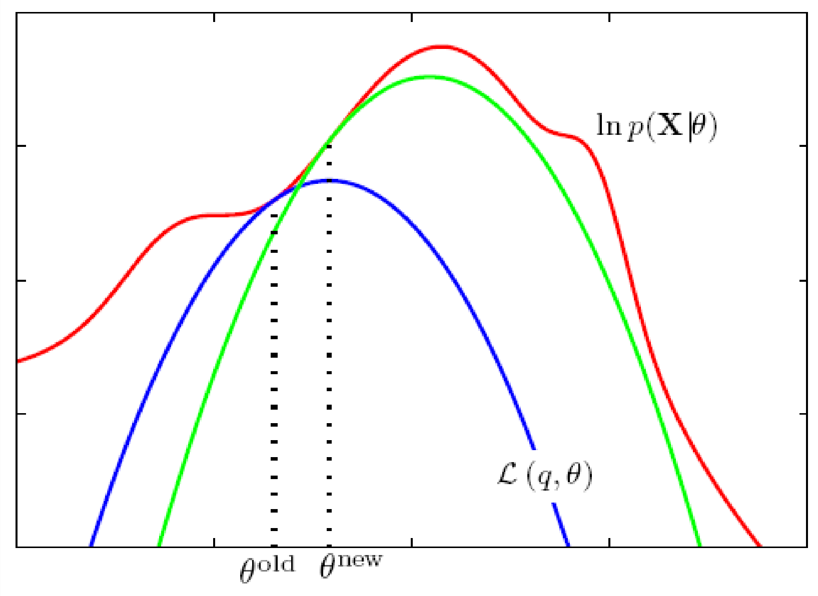
The EM algorithm: Gaussian Mixture

- The algorithm converges pretty fast after a few iterations: $(\widehat{\mu}_1, \widehat{\mu}_2, \widehat{\pi})$

```
[1] -1.7424035  0.1277127  0.3877030
[1] -2.2467959  0.9673550  0.3825091
[1] -2.2117884  1.3957797  0.3310913
[1] -2.1518538  1.6386121  0.2993035
[1] -2.1167579  1.7706276  0.2828132
[1] -2.0986018  1.8367258  0.2747542
[1] -2.0897518  1.8682925  0.2709414
[1] -2.0855753  1.8830238  0.2691684
[1] -2.0836364  1.8898248  0.2683511
[1] -2.0827434  1.8929490  0.2679758
[1] -2.0823335  1.8943810  0.2678039
```

The EM Algorithm for General Purpose

The EM Algorithm



The EM Algorithm

- Suppose that we want to maximize the a log-likelihood

$$\log \mathsf{p}(\mathbf{x}|\boldsymbol{\theta})$$

- Under some scenarios this likelihood can be difficult to derive:
  - $\mathbf{X}$ are generated from a mixture of several models, however, we do not know which is the underlying true model for each observation (GMM belongs to this case).
  - $\mathbf{X}$ contains missing values, where we have $\mathbf{X} = (\mathbf{X}_{\text{obs}}, \mathbf{X}_{\text{mis}})$.
- However, it would be easier if we introduce a latent variable $\mathbf{Z}$, such that the joint likelihood of $\mathsf{p}(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})$ is much easier to derive.

The EM algorithm

- For example, if $\mathbf{Z}$ represents the hidden label, then

$$\mathsf{p}(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) = \mathsf{p}(\mathbf{z}|\boldsymbol{\theta})\mathsf{p}(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$$

  where both probabilities are easier to write out given the underlying model.

- In general, for a discrete case of $\mathbf{Z}$, we need to maximize the log-likelihood

$$\log \mathsf{p}(\mathbf{x}|\boldsymbol{\theta}) = \log \sum_{\mathbf{z}} \mathsf{p}(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})$$

- For a continuous case, we maximize the log-likelihood

$$\log \mathsf{p}(\mathbf{x}|\boldsymbol{\theta}) = \log \int_{\mathbf{z}} \mathsf{p}(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) \mathsf{d}\mathbf{z}$$

The EM algorithm

- This can still be difficult to solve since there is a summation in the $\log$ function.
- The EM (Expectation–Maximization) algorithm is designed to solve this problem (Dempster, Laird, and Rubin, 1977)
- An EM algorithem consists of two steps:
  - E-step: Under the current value of $\boldsymbol{\theta}$, denoted as $\boldsymbol{\theta}^{(k)}$, find $p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^{(k)})$, the distribution of the unobserved variables given the data and $\boldsymbol{\theta}^{(k)}$. Then calculate the conditional expectation:

$$
\begin{aligned}
g(\boldsymbol{\theta}) &= \mathbb{E}_{\mathbf{Z}|\mathbf{x},\boldsymbol{\theta}^{(k)}} \log \mathsf{p}(\mathbf{x}, \mathbf{Z}|\boldsymbol{\theta}) \\
&= \begin{cases} \sum_{\mathbf{z}} \mathsf{p}(\mathbf{Z} = \mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^{(k)}) \log \mathsf{p}(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) & \text{(discrete)} \\ \int_{\mathbf{z}} \mathsf{p}(\mathbf{Z} = \mathbf{z}|\mathbf{x}, \boldsymbol{\theta}^{(k)}) \log \mathsf{p}(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) d\mathbf{z} & \text{(continuous)}. \end{cases}
\end{aligned}
$$

  - M-step: Re-estimate the parameter $\boldsymbol{\theta}$ to maximize $g(\boldsymbol{\theta})$:

$$
\boldsymbol{\theta}^{(k+1)} = \arg\max_{\boldsymbol{\theta}} \; g(\boldsymbol{\theta})
$$

Remarks

- Iterations will converge, but no guarantee converging to the MLE (likely stop at a local optium)
- To escape a local optium: random-restart hill climbing, simulated annealing
- The EM is especially useful when the likelihood is an exponential family: the E step becomes the sum of expectations of sufficient statistics, and the M step involves maximizing a linear function
- In general, the convergence rate of the EM algorithm is of the first order

Hidden Markov Models

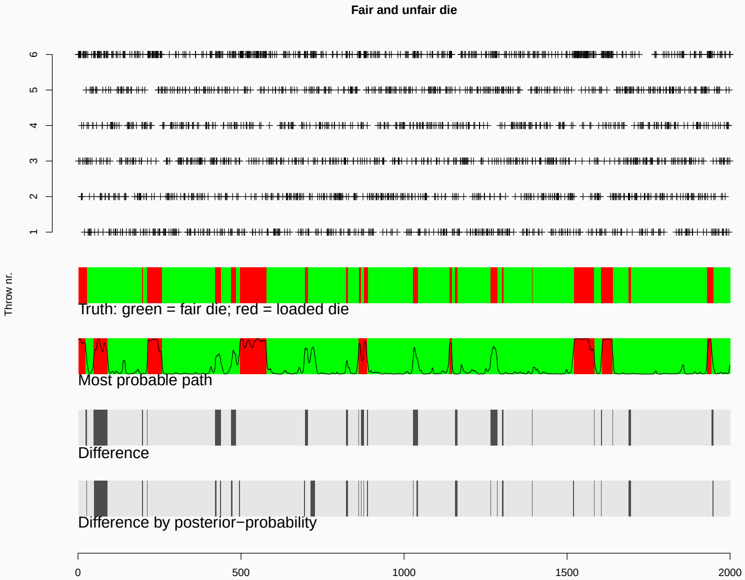
The Dishonest Casino Example

- An example taken from Durbin et. al. (1999).
- A dishonest casino uses two dice, one of them is fair and the other one is loaded.

Face/Prob	"1"	"2"	"3"	"4"	"5"	"6"
Fair Die	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$
Loaded Die	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{2}$

- The observer doesn't know which die is actually taken (the state is hidden), but the sequence of throws (observations) can be used to infer which die (state) was used.

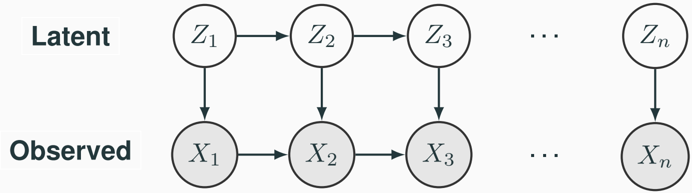
The Dishonest Casino Example

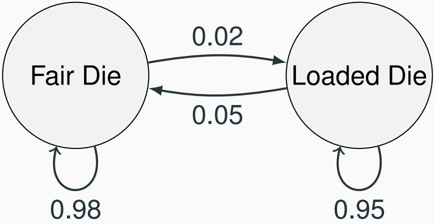


Hidden Markov Model

- Consider a Hidden Markov Model (HMM) for $(\mathbf{Z}, \mathbf{X}) = (Z_1, \ldots, Z_n, X_1, \ldots, X_n)$ where X_i's are observed (face of a dice) and Z_i's are hidden (fair or loaded). Let's assume that both $\mathbf{Z}$ and $\mathbf{X}$ are discrete random variables, taking m_z and m_x possible values, respectively. So the HMM is parameterized by $\theta = (w, A, B)$ where
    - $w_{m_z \times 1}$: distribution for Z_1, an initial stage.
    - $A_{m_z \times m_z}$: the transition probability matrix from Z_t to Z_{t+1}.
    - $B_{m_z \times m_x}$: the probability matrix (the emission distribution) for observing X_t under each hidden stage Z_t.
- A behavior of a HMM is fully determined by the three probabilities w, A, and B, and implicitly m_z and m_x.

Hidden Markov Model





Elements of a HMM

- For the Dishonest Casino Example, we have
  - $m_z = 2$, $m_x = 6$
  - $A = \begin{bmatrix} 0.98 & 0.02 \\ 0.05 & 0.95 \end{bmatrix}$
  - $B = \begin{bmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{10} & \frac{1}{10} & \frac{1}{10} & \frac{1}{10} & \frac{1}{10} & \frac{1}{2} \end{bmatrix}$
  - $w = \left(\frac{1}{2}, \frac{1}{2}\right)$, equal probabilities if no strong prior believe
- We can calculate the probabilities of the observed data based on any given parameter value.

Modeling the data

- How to model the data and detect the underlying states (which die was used)?
- The underlying states $\{Z_t, t = 1, \ldots, n\}$ is a markov chain, that satisfies the following assumptions:
- The memoryless assumption:

$$\mathsf{p}(Z_t | Z_{t-1}, \ldots, Z_1) = \mathsf{p}(Z_t | Z_{t-1})$$

- The stationary assumption:

$$\mathsf{p}(Z_t | Z_{t-1}) = \mathsf{p}(Z_2 | Z_1), \text{ for } t = 2, \ldots, n$$

Modeling the data

- The log-likelihood on the observed data is given by

$$\log\left[\mathsf{p}(\mathbf{x}|\boldsymbol{\theta})\right] = \log\left[\sum_{\mathbf{z}}\mathsf{p}(\mathbf{x},\mathbf{z}|\boldsymbol{\theta})\right]$$
$$= \log\left[\sum_{\mathbf{z}}\mathsf{p}(\mathbf{z}|\boldsymbol{\theta})\mathsf{p}(\mathbf{x}|\mathbf{z},\boldsymbol{\theta})\right]$$

  which is very hard to optimize due to the summation inside the log (generally not convex).

- Note: with a slight abuse of notation, here $\mathbf{x}$ and $\mathbf{z}$ are the observed vectors of the sequence of $\mathbf{X}$ and $\mathbf{Z}$.
- The the Baum-Welch algorithm is developed to solve this problem. It uses the EM algorithm and the forward-backward algorithm.

Appendix: Forward-Backward Algorithm for HMM

EM algorithm:

- E-step: Under the current value of θ, denoted as $\theta^{(k)}$, find $\mathsf{p}(\mathbf{z}|\mathbf{x}, \theta^{(k)})$, the distribution of the unobserved variables given the observed data and $\theta^{(k)}$. Then calculate:

$$g(\theta) = \mathbb{E}_{\mathbf{Z}|\mathbf{x},\theta^{(k)}} \log \mathsf{p}(\mathbf{x}, \mathbf{Z}|\theta)$$
$$= \begin{cases} \sum_{\mathbf{z}} \mathsf{p}(\mathbf{Z} = \mathbf{z}|\mathbf{x}, \theta^{(k)}) \log \mathsf{p}(\mathbf{x}, \mathbf{z}|\theta) & \text{(discrete)} \\ \int_{\mathbf{z}} \mathsf{p}(\mathbf{Z} = \mathbf{z}|\mathbf{x}, \theta^{(k)}) \log \mathsf{p}(\mathbf{x}, \mathbf{z}|\theta) d\mathbf{z} & \text{(continuous)}. \end{cases}$$

- M-step: Re-estimate the parameter θ to maximize $g(\theta)$:

$$\theta^{(k+1)} = \arg\max_{\theta} \; g(\theta)$$

- How to calculate $\mathsf{p}(\mathbf{z}|\mathbf{x}, \theta^{(k)})$ for our HMM problem?

EM algorithm

- To calculate the E-step, we first write out the log-likelihood of the complete data (recall the memoryless and stationary assumptions):

$$\log \mathsf{p}(\mathbf{z}, \mathbf{x}|\theta) = \log w(z_1) + \sum_{t=1}^{n-1} \log A(z_t, z_{t+1}) + \sum_{t=1}^{n} \log B(z_t, x_t),$$

  and then try to integrate it over all possible values of $\mathbf{Z}$, based on a current "guess", $\theta^{(k)}$:

$$\mathbb{E}_{\mathbf{Z}|\mathbf{x},\theta^{(k)}} \log \mathsf{p}(\mathbf{x}, \mathbf{Z}|\theta)$$

- To calculate this expectation, we need the conditional distribution of $Z|X, \theta^{(k)}$, which is just the conditional expectations:

$$\gamma_t(i, j) = \mathsf{p}(Z_t = i, Z_{t+1} = j|\mathbf{x}, \theta^{(k)})$$
$$\gamma_t(i) = \mathsf{p}(Z_t = i|\mathbf{x}, \theta^{(k)})$$

EM algorithm

- Suppose we already have the γ_t values, the E-step is:

$$
\begin{aligned}
&\mathbb{E}_{\mathbf{Z}|\mathbf{x},\theta^{(k)}} \log p(\mathbf{x}, \mathbf{Z}|\theta) \\
&= \mathbb{E}_{\mathbf{Z}|\mathbf{x},\theta^{(k)}} \left[\log w(Z_1) + \sum_{t=1}^{n-1} \log A(Z_t, Z_{t+1}) + \sum_{t=1}^{n} \log B(Z_t, x_t) \right] \\
&= \sum_{i=1}^{m_z} \gamma_1(i) \log w(i) + \sum_{t=1}^{n-1} \sum_{i,j=1}^{m_z} \gamma_t(i,j) \log A(i,j) \\
&\qquad + \sum_{t=1}^{n} \sum_{i}^{m_z} \gamma_t(i) \log B(i, x_t)
\end{aligned}
$$

- If we can compute each $\gamma_t(i,j)$ and $\gamma_t(i)$, this step is done.

EM algorithm

- At the M-step, we update the parameters $\theta = (w, A, B)$

$$
\begin{aligned}
w^{(k+1)}(i) &= \gamma_1(i), && i = 1, \ldots, m_z; \\
A^{(k+1)}(i,j) &= \frac{\sum_{t=1}^{n-1} \gamma_t(i,j)}{\sum_{j'} \sum_{t=1}^{n-1} \gamma_t(i,j')}, && i,j = 1, \ldots, m_z; \\
B^{(k+1)}(i,l) &= \frac{\sum_{t:x_t=l} \gamma_t(i)}{\sum_{t=1}^{n} \gamma_t(i)}, && i = 1, \ldots, m_z,\ l = 1, \ldots, m_x.
\end{aligned}
$$

- They are just the MLE estimators obtained by pooling and averaging a particular transition/emission event.
- For example, $B(i,l)$ is observing $X = l$ if the state is $Z = i$, so we go through all events with $Z = i$ in the entire chain, and average the events where $X = l$ is observed to get the probability.

Forward-Backward Algorithm

- There is still a remaining difficulty for calculating the conditional probabilities

$$\gamma_t(i, j) = \mathsf{p}(Z_t = i, Z_{t+1} = j | \mathbf{x}, \theta^{(k)}),$$

  the conditional probability of moving from state i to state j at time point t given all the observed data $\mathbf{x}$, and

$$\gamma_t(i) = \mathsf{p}(Z_t = i | \mathbf{x}, \theta^{(k)}),$$

  the conditional probability of being at state i at time point t given all the observed data $\mathbf{x}$.
- We will use a forward-backward algorithm to calculate this.

Forward-Backward Algorithm

- With no risk of ambiguity, we will omit $\theta^{(k)}$ from the notation, i.e., p is in fact $\mathsf{p}_{\theta^{(k)}}$
- For $\gamma_t(i,j)$, by Bayes' theorem, we have

$$\begin{aligned}
\gamma_t(i,j) &= \mathsf{p}(Z_t = i, Z_{t+1} = j | \mathbf{x}) \\
&\propto \mathsf{p}(\mathbf{x}_{1:t}, Z_t = i, Z_{t+1} = j, x_{t+1}, \mathbf{x}_{(t+2):n}) \\
&= \underbrace{\mathsf{p}(\mathbf{x}_{1:t}, Z_t = i)}_{\alpha_t(i)} \times \underbrace{\mathsf{p}(Z_{t+1} = j | Z_t = i)}_{A(i,j)} \\
&\quad \times \underbrace{\mathsf{p}(x_{t+1} | Z_{t+1} = j)}_{B(j,x_{t+1})} \times \underbrace{\mathsf{p}(\mathbf{x}_{(t+2):n} | Z_{t+1} = j)}_{\beta_{t+1}(j)} \quad \text{(why?)} \\
&\triangleq \alpha_t(i) A(i,j) B(j, x_{t+1}) \beta_{t+1}(j)
\end{aligned}$$

Forward-Backward Algorithm

- $\alpha_t(i) = \mathrm{p}(\mathbf{x}_{1:t}, Z_t = i)$ is the forward probability of observing $\mathbf{x}_{1:t}$ and having state i at time t;
- $\beta_{t+1}(j) = \mathrm{p}(\mathbf{x}_{(t+2):n} | Z_{t+1} = j)$ is the backward probability of observing $\mathbf{x}_{(t+2):n}$ given state j at time t.
- Note: $\alpha_t(i)$ is a joint probability, and $\beta_{t+1}(j)$ is a conditional probability.
- How to calculate $\alpha_t(i)$ and $\beta_{t+1}(j)$? We do this recursively starting from the two end points $t = 1$ and $t = n$.

Forward Probability

- For the first time point $t = 1$:

$$\alpha_1(i) = \mathsf{p}(x_1, Z_1 = i) = w(i)B(i, x_1),$$

- For each t, we can then calculate the next time point $\alpha_{t+1}(i)$ using the information of $\alpha_t(i)$:

$$\begin{aligned}
\alpha_{t+1}(i) &= \mathsf{p}(x_1, \ldots, x_{t+1}, Z_{t+1} = i) \\
&= \sum_j \mathsf{p}(x_1, \ldots, x_{t+1}, Z_t = j, Z_{t+1} = i) \\
&\quad \text{(exhaust all states of } Z_t \text{ in the previous } t\text{)} \\
&= \sum_j \mathsf{p}(\mathbf{x}_{1:t}, Z_t = j)\mathsf{p}(Z_{t+1} = i | Z_t = j)\mathsf{p}(x_{t+1} | Z_{t+1} = i) \\
&= \sum_j \alpha_t(j)A(j, i)B(i, x_{t+1})
\end{aligned}$$

Backward Probability

- For the last time point $t = n$, $\beta_n(i) = \mathsf{p}(\mathbf{x}_{n+1}|Z_n = i)$, but we don't have $\mathbf{x}_{n+1}$ (no information). Hence, to not inject any artificial information, we should let

$$\beta_n(i) = 1$$

- Then we recursively calculate the $\beta_{t-1}(i)$ in the previous state using $\beta_t(i)$:

$$\begin{aligned}
\beta_{t-1}(i) &= \mathsf{p}(x_t, \ldots, x_n | Z_{t-1} = i) \\
&= \sum_j \mathsf{p}(x_t, \ldots, x_n, Z_t = j | Z_{t-1} = i) \\
&\quad \text{(exhaust all states of } Z_t \text{ in the next } t\text{)} \\
&= \sum_j \mathsf{p}(Z_t = j | Z_{t-1} = i)\mathsf{p}(x_t | Z_t = j)\mathsf{p}(\mathbf{x}_{t+1:n} | Z_t = j) \\
&= \sum_j A(i,j)B(j, x_t)\beta_t(j)
\end{aligned}$$

Forward-Backward Algorithm

- The conditional probability $\gamma_t(i,j)$ needs to be normalized by the marginal probability to be a proper distribution:

$$\gamma_t(i,j) = \frac{\alpha_t(i)A(i,j)B(j,x_{t+1})\beta_{t+1}(j)}{\sum_i \sum_j \alpha_t(i)A(i,j)B(j,x_{t+1})\beta_{t+1}(j)}$$

- Similarly, we can calculate $\gamma_t(i)$ using Bayes' Theorem

$$\begin{aligned}
\gamma_t(i) &= \mathsf{p}(Z_t = i|\mathbf{x}) \\
&\propto \mathsf{p}(Z_t = i, \mathbf{x}) \\
&= \mathsf{p}(Z_t = i, \mathbf{x}_{1:t})\mathsf{p}(\mathbf{x}_{t+1:n}|Z_t = i, \mathbf{x}_{1:t}) \\
&= \mathsf{p}(Z_t = i, \mathbf{x}_{1:t})\mathsf{p}(\mathbf{x}_{t+1:n}|Z_t = i) \\
&= \alpha_t(i)\beta_t(i)
\end{aligned}$$

- Hence, after normalization

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_i \alpha_t(i)\beta_t(i)}$$