Hidden Markov Models

Filtering, Smoothing & Parameter Estimation

Sam Bowyer

Bootcamp Talk

7th December 2022

Table of Contents

Recap: Markov Chains

A sequence of random variables $X_1, X_2, \ldots$ taking values in a state space S is a Markov chain if it satisfies the Markov property $\forall t$:

$$\mathbb{P}(X_t = x_t | X_1 = x_1, \ldots, X_{t-1} = x_{t-1}) = \mathbb{P}(X_t = x_t | X_{t-1} = x_{t-1}).$$

That is, the value of X_t depends **only** on the value of X_{t-1}.

Markov Chains

Assume $S = \{1, 2, ..., N\} = [N]$ and that the Markov chain is time homogeneous:

$$\mathbb{P}(X_t = j | X_{t-1} = i) = \mathbb{P}(X_{t'} = j | X_{t'-1} = i) \;\; \forall t, t'.$$

Markov Chains

Assume $S = \{1, 2, ..., N\} = [N]$ and that the Markov chain is time homogeneous:

$$\mathbb{P}(X_t = j | X_{t-1} = i) = \mathbb{P}(X_{t'} = j | X_{t'-1} = i) \ \ \forall t, t'.$$

We can represent the Markov chain by (π, A) where:

- $\pi \in \mathbb{R}^N$ is the initial distribution over state space S:

$$\pi_i = \mathbb{P}(X_1 = i).$$

Markov Chains

Assume $S = \{1, 2, ..., N\} = [N]$ and that the Markov chain is time homogeneous:

$$\mathbb{P}(X_t = j | X_{t-1} = i) = \mathbb{P}(X_{t'} = j | X_{t'-1} = i) \;\; \forall t, t'.$$

We can represent the Markov chain by (π, A) where:

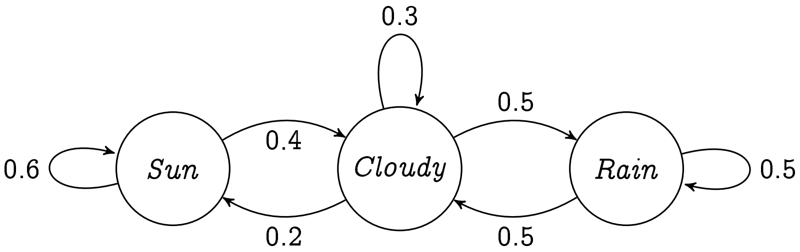
- $\pi \in \mathbb{R}^N$ is the initial distribution over state space S:

$$\pi_i = \mathbb{P}(X_1 = i).$$

- $A \in \mathbb{R}^{N \times N}$ gives us the transition probabilities:

$$a_{ij} = \mathbb{P}(X_t = j | X_{t-1} = i).$$

Markov Chains: Example



$$S = \{Sun, Cloudy, Rain\}, \quad A = \begin{pmatrix} 0.6 & 0.4 & 0 \\ 0.2 & 0.3 & 0.5 \\ 0 & 0.5 & 0.5 \end{pmatrix}$$

Hidden Markov Models

A hidden Markov Model involves an unobservable Markov chain $X_1, X_2, \ldots$ and a sequence of observations $Y_1, Y_2, \ldots$ such that:

$$\mathbb{P}(Y_t = y_t | X_1 = x_1, \ldots, X_t = x_t, Y_1 = y_1, \ldots, Y_{t-1} = y_{t-1}, Y_{t+1} = y_{t+1}, \ldots) \\ = \mathbb{P}(Y_t = y_t | X_t = x_t).$$

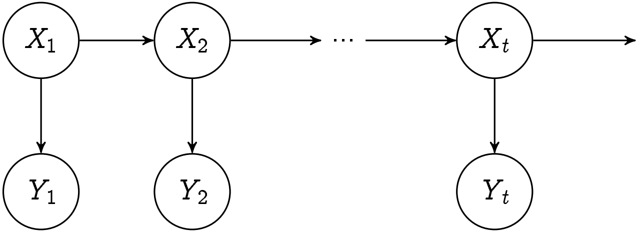
That is, Y_t depends **only** on X_t.

Hidden Markov Models

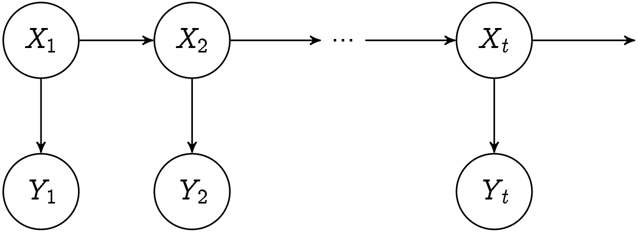
A hidden Markov Model involves an unobservable Markov chain $X_1, X_2, \ldots$ and a sequence of observations $Y_1, Y_2, \ldots$ such that:

$$\mathbb{P}(Y_t = y_t | X_1 = x_1, \ldots, X_t = x_t, Y_1 = y_1, \ldots, Y_{t-1} = y_{t-1}, Y_{t+1} = y_{t+1}, \ldots) = \mathbb{P}(Y_t = y_t | X_t = x_t).$$

That is, Y_t depends **only** on X_t.



Hidden Markov Models

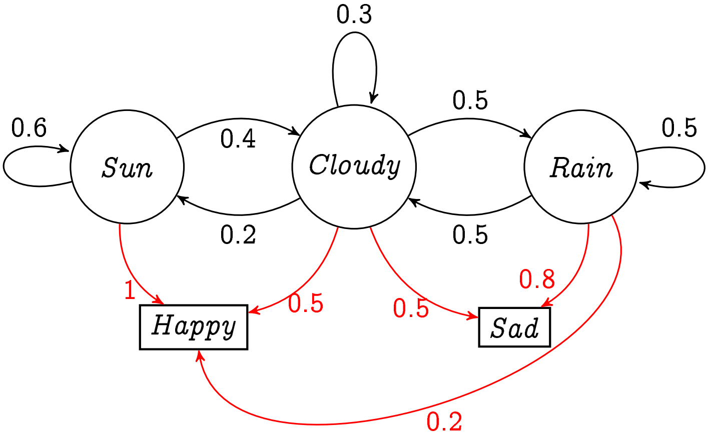


We suppose each Y_t takes values from a set of M possible observations $O = \{o_1, o_2, \ldots, o_M\}$, with emission/observation probabilities:

$$b_j(o_k) = \mathbb{P}(Y_t = o_k | X_t = j)$$

for $k \in [M], j \in [N]$.

Hidden Markov Models: Example



With $O = \{Happy, Sad\}$:

- $b_{Sun}(Happy) = 1$
- $b_{Sun}(Sad) = 0$
- $b_{Cloudy}(Happy) = 0.5$
- $b_{Cloudy}(Sad) = 0.5$
- $b_{Rain}(Happy) = 0.2$
- $b_{Rain}(Sad) = 0.8$

Hidden Markov Models

An HHM with hidden state space $S = [N]$ and observation space $O = \{o_1, ..., o_M\}$ can be parameterised fully as $\lambda = (\pi, A, B)$ where:

- $\pi \in \mathbb{R}^N$ is the initial distribution over state space S:

$$\pi_i = \mathbb{P}(X_1 = i).$$

Hidden Markov Models

An HHM with hidden state space $S = [N]$ and observation space $O = \{o_1, ..., o_M\}$ can be parameterised fully as $\lambda = (\pi, A, B)$ where:

- $\pi \in \mathbb{R}^N$ is the initial distribution over state space S:

$$\pi_i = \mathbb{P}(X_1 = i).$$

- $A \in \mathbb{R}^{N \times N}$ gives us the transition probabilities:

$$a_{ij} = \mathbb{P}(X_t = j | X_{t-1} = i).$$

Hidden Markov Models

An HHM with hidden state space $S = [N]$ and observation space $O = \{o_1, \ldots, o_M\}$ can be parameterised fully as $\lambda = (\pi, A, B)$ where:

- $\pi \in \mathbb{R}^N$ is the initial distribution over state space S:

$$\pi_i = \mathbb{P}(X_1 = i).$$

- $A \in \mathbb{R}^{N \times N}$ gives us the transition probabilities:

$$a_{ij} = \mathbb{P}(X_t = j | X_{t-1} = i).$$

- $B = \{b_j(o_k) : j \in [N], k \in [M]\}$ gives us the emission/observation probabilities:

$$b_j(o_k) = \mathbb{P}(Y_t = o_k | X_t = j).$$

Hidden Markov Models: Applications

- Speech Recognition (e.g. [Rabiner, 1989]):
  - Hidden states: basic parts of speech (e.g. words, syllables, phonemes etc.).
  - Observations: sections of audio signal.

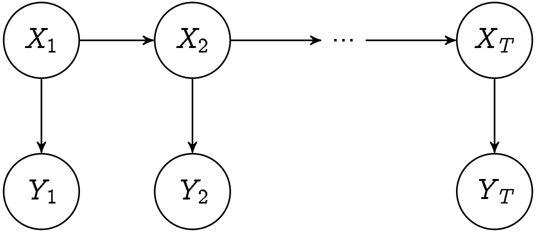
Hidden Markov Models: Applications

- Speech Recognition (e.g. [Rabiner, 1989]):
  - Hidden states: basic parts of speech (e.g. words, syllables, phonemes etc.).
  - Observations: sections of audio signal.

- Bioinformatics (e.g. [Wong et al., 2013]):
  - Hidden states: sequences of nucleotides (A, T, C and G) within a DNA sequence.
  - Observations: intensity of chemical reaction when testing a protein.

HMMs: Important Questions

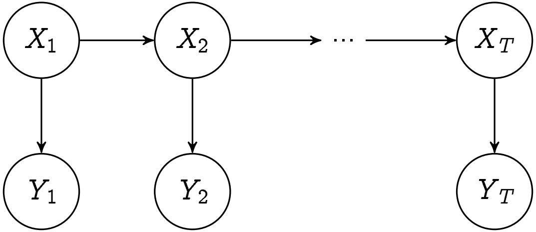
Given an HHM $\lambda = (\pi, A, B)$ and a sequence of observations $Y = (Y_1, ..., Y_T)$:



Q1. What is $\mathbb{P}(Y|\lambda)$?

HMMs: Important Questions

Given an HHM $\lambda = (\pi, A, B)$ and a sequence of observations $Y = (Y_1, ..., Y_T)$:



Q1. What is $\mathbb{P}(Y|\lambda)$?

Q2. What value is X_t likely to have taken for any $t \in [T]$?

A Naïve Approach To Q1

The probability of observing $Y = (Y_1, ..., Y_T)$ given an underlying sequence of states $X = (X_1, ..., X_T)$:

$$\mathbb{P}(Y|X, \lambda) = \prod_{t=1}^{T} b_{X_t}(Y_t).$$

A Naïve Approach To Q1

The probability of observing $Y = (Y_1, \ldots, Y_T)$ given an underlying sequence of states $X = (X_1, \ldots, X_T)$:

$$\mathbb{P}(Y|X, \lambda) = \prod_{t=1}^{T} b_{X_t}(Y_t).$$

The probability of λ producing $X = (X_1, \ldots, X_T)$:

$$\mathbb{P}(X|\lambda) = \pi_{X_1} \prod_{t=2}^{T} a_{X_{t-1}, X_t}.$$

A Naïve Approach To Q1

The probability of observing $Y = (Y_1, ..., Y_T)$ given an underlying sequence of states $X = (X_1, ..., X_T)$:

$$\mathbb{P}(Y|X,\lambda) = \prod_{t=1}^{T} b_{X_t}(Y_t).$$

The probability of λ producing $X = (X_1, ..., X_T)$:

$$\mathbb{P}(X|\lambda) = \pi_{X_1} \prod_{t=2}^{T} a_{X_{t-1},X_t}.$$

Hence, marginalising over all possible X

$$\mathbb{P}(Y|\lambda) = \sum_{X} \mathbb{P}(Y|X,\lambda)\mathbb{P}(X|\lambda) = \sum_{X=(X_1,...,X_T)} \pi_{X_1} b_{X_1}(Y_1) \prod_{t=2}^{T} a_{X_{t-1},X_t} b_{X_t}(Y_t).$$

A Naïve Approach To Q1

$$\mathbb{P}(Y|\lambda) = \sum_{X=(X_1,\dots,X_T)} \pi_{X_1} b_{X_1}(Y_1) \prod_{t=2}^{T} a_{X_{t-1},X_t} b_{X_t}(Y_t).$$

- Each summand involves multiplying $2T$ terms together.

A Naïve Approach To Q1

$$\mathbb{P}(Y|\lambda) = \sum_{X=(X_1,\ldots,X_T)} \pi_{X_1} b_{X_1}(Y_1) \prod_{t=2}^{T} a_{X_{t-1},X_t} b_{X_t}(Y_t).$$

- Each summand involves multiplying $2T$ terms together.
- There are N^T possible values for X.

A Naïve Approach To Q1

$$\mathbb{P}(Y|\lambda) = \sum_{X=(X_1,\ldots,X_T)} \pi_{X_1} b_{X_1}(Y_1) \prod_{t=2}^{T} a_{X_{t-1},X_t} b_{X_t}(Y_t).$$

- Each summand involves multiplying $2T$ terms together.
- There are N^T possible values for X.
- Hence this calculation has $\mathcal{O}(TN^T)$ complexity.

A Naïve Approach To Q1

$$\mathbb{P}(Y|\lambda) = \sum_{X=(X_1,\ldots,X_T)} \pi_{X_1} b_{X_1}(Y_1) \prod_{t=2}^{T} a_{X_{t-1},X_t} b_{X_t}(Y_t).$$

- Each summand involves multiplying $2T$ terms together.
- There are N^T possible values for X.
- Hence this calculation has $\mathcal{O}(TN^T)$ complexity.
- We can do much better if we utilise recursion.

Filtering

Predict X_t based on $Y_1, \ldots, Y_t$.

For $i \in [N]$, $t \in [T]$ let our *forward* probabilities be:

$$\alpha_t(i) = \mathbb{P}(Y_1, \ldots, Y_t, X_t = i | \lambda).$$

Filtering

Predict X_t based on $Y_1, \ldots, Y_t$.

For $i \in [N]$, $t \in [T]$ let our *forward* probabilities be:

$$\alpha_t(i) = \mathbb{P}(Y_1, \ldots, Y_t, X_t = i|\lambda).$$

Calculate inductively:

$$\alpha_1(i) = \pi_i b_i(Y_1)$$

Filtering

Predict X_t based on $Y_1, \ldots, Y_t$.
For $i \in [N]$, $t \in [T]$ let our *forward* probabilities be:

$$\alpha_t(i) = \mathbb{P}(Y_1, \ldots, Y_t, X_t = i | \lambda).$$

Calculate inductively:

$$\alpha_1(i) = \pi_i b_i(Y_1)$$

$$\alpha_{t+1}(j) = \underbrace{\left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right]}_{\text{All of the ways to get to state } j \text{ from any state } i \text{ at time step } t} \cdot b_j(Y_{t+1})$$

Q1 Using Forward Probabilities

We can now calculate $\mathbb{P}(Y|\lambda)$ as:

$$\mathbb{P}(Y|\lambda) = \sum_{i=1}^{N} \alpha_T(i).$$

Q1 Using Forward Probabilities

We can now calculate $\mathbb{P}(Y|\lambda)$ as:

$$\mathbb{P}(Y|\lambda) = \sum_{i=1}^{N} \alpha_T(i).$$

- Calculation of $\alpha_1(i)\ \forall i$ is $\mathcal{O}(N)$.

Q1 Using Forward Probabilities

We can now calculate $\mathbb{P}(Y|\lambda)$ as:

$$\mathbb{P}(Y|\lambda) = \sum_{i=1}^{N} \alpha_T(i).$$

- Calculation of $\alpha_1(i)\ \forall i$ is $\mathcal{O}(N)$.
- Given $\alpha_t(i)\ \forall i$, calculation of $\alpha_{t+1}(j)\ \forall j$ is $\mathcal{O}(N^2)$.

Q1 Using Forward Probabilities

We can now calculate $\mathbb{P}(Y|\lambda)$ as:

$$\mathbb{P}(Y|\lambda) = \sum_{i=1}^{N} \alpha_T(i).$$

- Calculation of $\alpha_1(i)\ \forall i$ is $\mathcal{O}(N)$.
- Given $\alpha_t(i)\ \forall i$, calculation of $\alpha_{t+1}(j)\ \forall j$ is $\mathcal{O}(N^2)$.
  - (We calculate the probability of an i-to-j transition for every possible (i, j) pair.)

Q1 Using Forward Probabilities

We can now calculate $\mathbb{P}(Y|\lambda)$ as:

$$\mathbb{P}(Y|\lambda) = \sum_{i=1}^{N} \alpha_T(i).$$

- Calculation of $\alpha_1(i)\ \forall i$ is $\mathcal{O}(N)$.
- Given $\alpha_t(i)\ \forall i$, calculation of $\alpha_{t+1}(j)\ \forall j$ is $\mathcal{O}(N^2)$.
  - (We calculate the probability of an i-to-j transition for every possible (i, j) pair.)
- Calculating $\mathbb{P}(Y|\lambda)$ requires summing T values hence the overall complexity is $\mathcal{O}(TN^2)$.

Q2 Using Forward Probabilities: Filtering

Having these forward probabilities allows us to calculate $\mathbb{P}(X_t | Y_1, \ldots, Y_t, \lambda)$:

$$\mathbb{P}(X_t | Y_1, \ldots, Y_t, \lambda) = \frac{\mathbb{P}(X_t, Y_1, \ldots, Y_t | \lambda)}{\mathrm{P}(Y_1, \ldots, Y_t | \lambda)} = \frac{\alpha_t(X_t)}{\sum_{i=1}^{N} \alpha_T(i)}.$$

Q2 Using Forward Probabilities: Filtering

Having these forward probabilities allows us to calculate $\mathbb{P}(X_t | Y_1, ..., Y_t, \lambda)$:

$$\mathbb{P}(X_t | Y_1, ..., Y_t, \lambda) = \frac{\mathbb{P}(X_t, Y_1, ..., Y_t | \lambda)}{\mathrm{P}(Y_1, ..., Y_t | \lambda)} = \frac{\alpha_t(X_t)}{\sum_{i=1}^{N} \alpha_T(i)}.$$

This procedure is known as the Forward Algorithm.

Q2 Using Forward Probabilities: Filtering

Having these forward probabilities allows us to calculate $\mathbb{P}(X_t | Y_1, \ldots, Y_t, \lambda)$:

$$\mathbb{P}(X_t | Y_1, \ldots, Y_t, \lambda) = \frac{\mathbb{P}(X_t, Y_1, \ldots, Y_t | \lambda)}{\mathrm{P}(Y_1, \ldots, Y_t | \lambda)} = \frac{\alpha_t(X_t)}{\sum_{i=1}^{N} \alpha_T(i)}.$$

This procedure is known as the Forward Algorithm.

We can also get maximum a posteriori (MAP) estimates:

$$X_t^{\text{MAP}} = \operatorname*{argmax}_i \frac{\alpha_t(i)}{\sum_{j=1}^{N} \alpha_T(j)} = \operatorname*{argmax}_i \alpha_t(i)$$

Q2 Using Forward Probabilities: Filtering

Having these forward probabilities allows us to calculate $\mathbb{P}(X_t | Y_1, ..., Y_t, \lambda)$:

$$\mathbb{P}(X_t | Y_1, ..., Y_t, \lambda) = \frac{\mathbb{P}(X_t, Y_1, ..., Y_t|\lambda)}{\mathrm{P}(Y_1, ..., Y_t|\lambda)} = \frac{\alpha_t(X_t)}{\sum_{i=1}^{N} \alpha_T(i)}.$$

This procedure is known as the Forward Algorithm.

We can also get maximum a posteriori (MAP) estimates:

$$X_t^{\mathrm{MAP}} = \underset{i}{\mathrm{argmax}} \frac{\alpha_t(i)}{\sum_{j=1}^{N} \alpha_T(j)} = \underset{i}{\mathrm{argmax}}\, \alpha_t(i)$$

Smoothing: can we improve this by also using $Y_{t+1}, ..., Y_T$?

Smoothing: Backwards Probabilities

For each $i \in [N]$ and $t \in [T]$ we define

$$\beta_t(i) = \mathbb{P}(Y_{t+1}, \ldots, Y_T | X_t = i, \lambda)$$

Smoothing: Backwards Probabilities

For each $i \in [N]$ and $t \in [T]$ we define

$$\beta_t(i) = \mathbb{P}(Y_{t+1}, \ldots, Y_T | X_t = i, \lambda)$$

and calculate these inductively:

$$\beta_T(i) = 1$$

Smoothing: Backwards Probabilities

For each $i \in [N]$ and $t \in [T]$ we define

$$\beta_t(i) = \mathbb{P}(Y_{t+1}, \ldots, Y_T | X_t = i, \lambda)$$

and calculate these inductively:

$$\beta_T(i) = 1$$

$$\beta_t(i) = \underbrace{\sum_{j=1}^{N} a_{ij} b_j(Y_{t+1}) \beta_{t+1}(j)}_{\substack{\text{All of the ways to get to some state } j \\ \text{from state } i \text{ and observe } Y_{t+1}}}$$

Smoothing: Forward-Backward Algorithm

How can we combine the information given by Y_1, ..., Y_t (via $\alpha_t(i)$) and by Y_{t+1}, ..., Y_T (via $\beta_t(i)$)?

Smoothing: Forward-Backward Algorithm

How can we combine the information given by $Y_1, \ldots, Y_t$ (via $\alpha_t(i)$) and by $Y_{t+1}, \ldots, Y_T$ (via $\beta_t(i)$)?

$$\gamma_t(i) = \mathbb{P}(X_t = i | Y, \lambda) = \frac{\mathbb{P}(X_t = i, Y_1, \ldots, Y_t | \lambda)\mathbb{P}(Y_{t+1}, \ldots, Y_T | X_t = i, \lambda)}{\mathbb{P}(Y|\lambda)}$$

$$= \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

Smoothing: Forward-Backward Algorithm

How can we combine the information given by $Y_1, \ldots, Y_t$ (via $\alpha_t(i)$) and by $Y_{t+1}, \ldots, Y_T$ (via $\beta_t(i)$)?

$$\gamma_t(i) = \mathbb{P}(X_t = i | Y, \lambda) = \frac{\mathbb{P}(X_t = i, Y_1, \ldots, Y_t | \lambda)\mathbb{P}(Y_{t+1}, \ldots, Y_T | X_t = i, \lambda)}{\mathbb{P}(Y|\lambda)}$$

$$= \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}$$

We can now get improved MAP estimates: $X_t^{\text{MAP}} = \text{argmax}_i \, \gamma_t(i)$.

Smoothing: Forward-Backward Algorithm

We can now get improved MAP estimates:

$$X_t^{\text{MAP}} = \operatorname*{argmax}_i \gamma_t(i).$$

NOTE: X_1^{MAP}, ..., X_T^{MAP} is not necessarily the most probable sequence of X_1, ..., X_T given Y; it might even include impossible transitions from X_t to X_{t+1}.

Smoothing: Forward-Backward Algorithm

We can now get improved MAP estimates:

$$X_t^{\text{MAP}} = \operatorname*{argmax}_i \gamma_t(i).$$

NOTE: $X_1^{\text{MAP}}, \ldots, X_T^{\text{MAP}}$ is not necessarily the most probable sequence of $X_1, \ldots, X_T$ given Y; it might even include impossible transitions from X_t to X_{t+1}.

To find the most probable sequence $X = X_1, \ldots, X_T$ that maximises $\mathbb{P}(X \mid Y, \lambda)$, we'd have to use other techniques (e.g. Viterbi algorithm).

Parameter Estimation

What if we don't know $\lambda = (\pi, A, B)$?

Parameter Estimation

What if we don't know $\lambda = (\pi, A, B)$?

Given an observed sequence $Y = Y_1, ..., Y_T$ we can estimate λ using the forward-backward algorithm's machinery via the Baum-Welch Algorithm.

Parameter Estimation: The Baum-Welch Algorithm

Choose some initial parameter values $\lambda = (\pi, A, B)$.

Parameter Estimation: The Baum-Welch Algorithm

Choose some initial parameter values $\lambda = (\pi, A, B)$.

Introduce $\xi_t(i,j)$ where:

$$\xi_t(i,j) = \mathbb{P}(X_t = i, X_{t+1} = j \mid Y, \lambda) = \frac{\overbrace{\alpha_t(i)}^{\substack{\text{being in state} \\ i \text{ at time } t}} \cdot \overbrace{a_{ij}\, b_j(Y_{t+1})}^{\substack{\text{moving from } i \text{ to } j \\ \text{and observing } Y_{t+1}}} \cdot \overbrace{\beta_{t+1}(j)}^{\substack{\text{being in state } j \\ \text{at time } t+1}}}{\mathbb{P}(Y \mid \lambda)}$$

$$= \frac{\alpha_t(i) \cdot a_{ij}\, b_j(Y_{t+1}) \cdot \beta_{t+1}(j)}{\sum_{k=1}^{N} \sum_{l=1}^{N} \alpha_t(k) \cdot a_{kl}\, b_l(Y_{t+1}) \cdot \beta_{t+1}(l)}$$

Parameter Estimation: The Baum-Welch Algorithm

Now we have four quantities to work with:

Parameter Estimation: The Baum-Welch Algorithm

Now we have four quantities to work with:

- $\alpha_t(i) = \mathbb{P}(Y_1, ..., Y_t, X_t = i | \lambda)$

Parameter Estimation: The Baum-Welch Algorithm

Now we have four quantities to work with:

- $\alpha_t(i) = \mathbb{P}(Y_1, ..., Y_t, X_t = i|\lambda)$
- $\beta_t(i) = \mathbb{P}(Y_{t+1}, ..., Y_T|X_t = i, \lambda)$

Parameter Estimation: The Baum-Welch Algorithm

Now we have four quantities to work with:

- $\alpha_t(i) = \mathbb{P}(Y_1, ..., Y_t, X_t = i|\lambda)$
- $\beta_t(i) = \mathbb{P}(Y_{t+1}, ..., Y_T|X_t = i, \lambda)$
- $\gamma_t(i) = \mathbb{P}(X_t = i|Y, \lambda)$

Parameter Estimation: The Baum-Welch Algorithm

Now we have four quantities to work with:

- $\alpha_t(i) = \mathbb{P}(Y_1, \ldots, Y_t, X_t = i | \lambda)$
- $\beta_t(i) = \mathbb{P}(Y_{t+1}, \ldots, Y_T | X_t = i, \lambda)$
- $\gamma_t(i) = \mathbb{P}(X_t = i | Y, \lambda)$
- $\xi_t(i, j) = \mathbb{P}(X_t = i, X_{t+1} = j | Y, \lambda)$

Parameter Estimation: The Baum-Welch Algorithm

Now we have four quantities to work with:

- $\alpha_t(i) = \mathbb{P}(Y_1, \ldots, Y_t, X_t = i|\lambda)$
- $\beta_t(i) = \mathbb{P}(Y_{t+1}, \ldots, Y_T|X_t = i, \lambda)$
- $\gamma_t(i) = \mathbb{P}(X_t = i|Y, \lambda)$
- $\xi_t(i, j) = \mathbb{P}(X_t = i, X_{t+1} = j|Y, \lambda)$

Parameter Estimation: The Baum-Welch Algorithm

Now we have four quantities to work with:

- $\alpha_t(i) = \mathbb{P}(Y_1, \ldots, Y_t, X_t = i | \lambda)$
- $\beta_t(i) = \mathbb{P}(Y_{t+1}, \ldots, Y_T | X_t = i, \lambda)$
- $\gamma_t(i) = \mathbb{P}(X_t = i | Y, \lambda)$
- $\xi_t(i, j) = \mathbb{P}(X_t = i, X_{t+1} = j | Y, \lambda)$

Note: $\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i, j)$.

Parameter Estimation: The Baum-Welch Algorithm

We have $\lambda = (\pi, A, B)$ but can we find an improved $\bar{\lambda} = (\bar{\pi}, \bar{A}, \bar{B})$?

Parameter Estimation: The Baum-Welch Algorithm

We have $\lambda = (\pi, A, B)$ but can we find an improved $\bar{\lambda} = (\bar{\pi}, \bar{A}, \bar{B})$?

$$\bar{\pi}_i = \mathbb{P}(X_1 = i|\lambda) = \gamma_1(i)$$

Parameter Estimation: The Baum-Welch Algorithm

We have $\lambda = (\pi, A, B)$ but can we find an improved $\bar{\lambda} = (\bar{\pi}, \bar{A}, \bar{B})$?

$$\bar{\pi}_i = \mathbb{P}(X_1 = i | \lambda) = \gamma_1(i)$$

$$\bar{a}_{ij} = \frac{\text{expected number of } i\text{-to-}j \text{ transitions}}{\text{expected number of } i\text{-to-}k \text{ transitions } \forall k} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

Parameter Estimation: The Baum-Welch Algorithm

We have $\lambda = (\pi, A, B)$ but can we find an improved $\bar{\lambda} = (\bar{\pi}, \bar{A}, \bar{B})$?

$$\bar{\pi}_i = \mathbb{P}(X_1 = i|\lambda) = \gamma_1(i)$$

$$\bar{a}_{ij} = \frac{\text{expected number of } i\text{-to-}j \text{ transitions}}{\text{expected number of } i\text{-to-}k \text{ transitions } \forall k} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$\bar{b_j}(o_k) = \frac{\text{expected number of } o_k \text{ observations from state } j}{\text{expected number of time steps in state } j} = \frac{\sum_{t=1}^{T} \mathbb{1}_{\{Y_t = o_k\}} \cdot \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

Parameter Estimation: The Baum-Welch Algorithm

The Baum-Welch algorithm is a form of expectation maximisation algorithm:

- Once we've found $\bar{\lambda} = (\bar{\pi}, \bar{A}, \bar{B})$ we repeat the procedure with these parameters instead of λ to find another set of (improved) parameters.
- We repeat until the parameters stabilise into a local optimum.

Conclusion

- HMMs give us a powerful and versatile way to describe complex processes.

Conclusion

- HMMs give us a powerful and versatile way to describe complex processes.
- Given an HHM $\lambda = (\pi, A, B)$ and a sequence of observations $Y = Y_1, \ldots, Y_T$ we can:

Conclusion

- HMMs give us a powerful and versatile way to describe complex processes.
- Given an HHM $\lambda = (\pi, A, B)$ and a sequence of observations $Y = Y_1, \ldots, Y_T$ we can:
  - Efficiently calculate $\mathbb{P}(Y|\lambda)$.

Conclusion

- HMMs give us a powerful and versatile way to describe complex processes.
- Given an HHM $\lambda = (\pi, A, B)$ and a sequence of observations $Y = Y_1, \ldots, Y_T$ we can:
  - Efficiently calculate $\mathbb{P}(Y|\lambda)$.
  - Estimate the states X_t, $1 \leqslant t \leqslant T$ through filtering and smoothing.

Conclusion

- HMMs give us a powerful and versatile way to describe complex processes.
- Given an HHM $\lambda = (\pi, A, B)$ and a sequence of observations $Y = Y_1, ..., Y_T$ we can:
  - Efficiently calculate $\mathbb{P}(Y|\lambda)$.
  - Estimate the states X_t, $1 \leqslant t \leqslant T$ through filtering and smoothing.
- Given only a sequence of observations $Y = Y_1, .., Y_T$ we can estimate $\lambda = (\pi, A, B)$ using the Baum-Welch algorithm.

References

Rabiner, L. (1989).
A tutorial on hidden Markov models and selected applications in speech recognition.
Proceedings of the IEEE, 77(2):257–286.

Wong, K.-C., Chan, T.-M., Peng, C., Li, Y., and Zhang, Z. (2013).
DNA motif elucidation using belief propagation.
Nucleic Acids Research, 41(16):e153.

Thank you

Any questions?