1 Goal:

Our goal is to estimate $\theta$ when we got the observations - Training set: $\{\vec{X}_1, \ldots, \vec{X}_j, \ldots, \vec{X}_N\}$, but we don’t have the set of states. That is to say, we look after $\theta$ that take the likelihood or the log likelihood (LL) to the maximum.

$$LL(\theta : \vec{D}) = \log(P(x^1, \ldots, x^N : \theta)) = \sum_{j=1}^{N} \log(P(x^j : \theta))$$

$\theta$ is set of the model’s parameters, and in our case we have two parameters types:

1) Emission: $e_k(\vec{X}_i)$
2) Transition: $\tau_{kl}$

2 Estimation with states set:

Suppose that the states set $\{S^1, \ldots, S^j, \ldots, S^N\}$, is given. We want to find the $\theta$ that maximize the log likelihood of the full data.

$$LL(\theta : \vec{D}^*) = \sum_{j=1}^{N} \sum_{i=1}^{n_j} \log(P(s^j_i : s^j_{i-1}) + P(x^j_i : s^j_i))$$

Two parameters will help us to estimate $\theta$:

1) Number of times of transition from one state to other, i.e. how many times we move from state $k$ to state $l$\footnote{1}

$$N_{kl} = \sum_{j=1}^{N} \sum_{i=1}^{n_j} I\{s^j_{i-1} = k, s^j_i = l\}$$

$$\hat{\tau}_{kl} = \frac{N_{kl}}{\sum_{l' \in \text{states}} N_{kl'}}$$

\footnote{1}I is refers to Indicator function: \url{https://en.wikipedia.org/wiki/Indicator_function}
2) For each state we want to know what letter it emits:

\[ N_{kx} = \sum_{j=1}^{N} \sum_{i=1}^{n_j} I\{s^d_i = k, x^d_i = x\} \]

\[ \hat{e}_n(x) = \frac{N_{kx}}{\sum_{x' \in \text{letters}} N_{kx'}} \]

### 2.1 Zero-Probability problem

If we never seen appearance of some letter in some state, we will calculate probability to zero. But, if in the future we will see that letter in this state, we will have a problem, because the probability will aim to infinity.

Therefore we will not give to nothing probability of 0 or 1, We will only define 0 < \( P(A) < 1 \) (except \( A \) is a logical expression).

In our case we will deal with it by using **pseudocount** (Attributed to Laplace) - We will add numbers to \( N_{kl} \) and \( N_{kx} \). Those numbers are determined before we have seen any observations.

Therefore we will add \( \alpha \) and \( \beta \) to our equations:

\[ N_{kl} = \sum_{j=1}^{N} \sum_{i=1}^{n_j} I\{s^d_{i-1} = k, s^d_i = l\} + \alpha \]

\[ N_{kx} = \sum_{j=1}^{N} \sum_{i=1}^{n_j} I\{s^d_i = k, x^d_i = x\} + \beta \]

#### 2.1.1 Example for pseudocount:

Suppose we flip a coin, and our observations are: \{H,H,H,T,T\}

Simply we can calculate MLE and say that \( P(H) = 0.6 \)

But if we estimate that coins have half-half distribution, Then we will suppose that before the real flips, we already have observations of 20 flips: 10 times Head, ans 10 times Tail. So, we will add to each side 10 appearance, and now the MLE will be: \( P(H) = \frac{13}{25} \sim 0.5 \)
3 Parameter Estimation:

Until now we worked with the assumption that we got the states. Now we want to estimate $\theta$ even we not have the states.

How we can do it?

- We can sample states and check if what we got is reasonable.
- We can optimize by the gradient - stochastic gradient descent.

The problem with those solutions that they not necessarily will bring us to the correct solution. The sample can be not strict, and the gradient can bring us to local maximum and not to the global maximum.

3.1 Baum - Welch solution:

We will see the solution of Baum-Welch, and we will use family of algorithms that known as EM - Expectation Maximization.
It is an iterative algorithm, that in any iteration it will solve two different problems: Expectation(ESP) and Maximization(MAX).

Possible stop conditions are:

1. Number of iterations is bigger from some threshold: $t > thr$
2. $thr > |\theta^{(t)} - \theta^{(t-1)}|$
3. $thr > |L(\theta^{(t)}) - L(\theta^{(t-1)})|$
3.1.1 Calculate Expected values

A)

\[ N_{kl}^{(t)} = E_{\theta^{(t)}}[N_{kl}] = \sum_{j=1}^{N} \sum_{i=1}^{n_j} E_{\theta^{(t)}}[I(s_{i-1} = k, s_i = l)] = \sum_{j=1}^{N} \sum_{i=1}^{n_j} P(s_{i-1} = k, s_i = l : x_j, \theta^{(t)}) \]

How we express the last phrase?

We already calculate \( P(s_i : x) \): \( P(s_i : x) = \frac{F_k(i - 1) * e_i^{(t)}(x_i) * \tau^{(t)}_{kl} * B_l^{(t)}(i)}{P(x)} \)

And now we get:

\[ N_{kl}^{(t)} = \sum_{j=1}^{N} \sum_{i=1}^{n_j} \frac{F_k(i - 1) * e_i^{(t)}(x_i) * \tau^{(t)}_{kl} * B_l^{(t)}(i)}{P(x)} \]

B)

\[ N_{kx}^{(t)} = E_{\theta^{(t)}}[N_{kx}] = \sum_{j=1}^{N} \sum_{i=1}^{n_j} E_{\theta^{(t)}}[I(s_i = k, x_i = x)] \]

We already have the observations, therefore we can take the \( x_i \) that really exist and not to all \( x \). So the last phrase can be written as \( P(s_i, x_i = x : x) \), and finally we will get:

\[ N_{kx}^{(t)} = \sum_{j=1}^{N} \sum_{i=1}^{n_j} P(s_i = k, x_i = x : \theta^{(t)}) = \sum_{j=1}^{N} \sum_{i=1}^{n_j} \frac{F_k(i) B_l(i)}{P(x)} \]

Of course we will add here pseudocount as well.

3.1.2 The Baum-Welch (BW) algorithm:

Init: \( \theta^{(t)} = \text{guess / arbitrary / random / NOT uniformy} \)

For (t=1:L):

E-step:

- forward: \( F_k(i) \rightarrow E_{\theta^{(t)}}[N_{kl} : \theta^{(t)}] \)
- backward: \( B_k(i) \rightarrow E_{\theta^{(t)}}[N_{kx} : \theta^{(t)}] \)
M-step:
\[ \theta^{(t+1)} : \]
\[ \tau_{kl} = \frac{E_{\theta^{(t)}}[N_{kl}]}{\sum_{l'} E_{\theta^{(t)}}[N_{kl'}]} \]
\[ e_k(x) = \frac{E_{\theta^{(t)}}[N_{kx}]}{\sum_{x'} E_{\theta^{(t)}}[N_{kx'}]} \]

Stop if:
\[ t > thr \]
\[ \text{Or} \quad thr > |\theta^{(t+1)} - \theta^{(t)}| \]
\[ \text{Or} \quad thr > |L(\theta^{(t+1)} : \overrightarrow{D}) - L(\theta^{(t)} : \overrightarrow{D})| \]

The algorithm convergent to local optimum, therfore we will run the algorithm several times for getting the global optimum.