Forward Probability , Backward Probability, Baum Welch Algorithm

HMM

HMM parameters are given by: $\Theta = (A, B, \Pi)$

HMM Definition:

- Set of states : S where |S|=N
- Start state : S_0 /*P(S_0)=1*/
- Output Alphabet : O where |O|=K
- State Transition Probabilities :

  A = {a_{ij}} (prob. of going from state S_i to state S_j)
- Emission Probabilities : B = {b_j (O_k)} (prob. Of outputting symbol O_k from state j)
- Initial State Probabilities : $\Pi = \{\Pi_i\} = \{P(X_1=i\}$

Baum Welch Algorithm

- When corpus is available but not labeled, will use EM algorithm to estimate parameters of HMM.
- EM algorithm is used to find the maximum likelihood estimate of the parameters of HMM.

EM Algorithm

1. Choose some initial values for $\Theta = (A, B, \Pi)$.
2. Repeat the following step until convergence.
3. Determine probable (state) paths:

   ……, $X_{t-1} = i$, $X_t = j$, ….
4. Count the expected number of transitions, $\{a_{ij}\}$, as well as the expected number of times various emissions, $b_j(O_k)$, are made.
5. Re-estimate $\Theta = (A, B, \Pi)$ using $\{a_{ij}\}$ and $b_j(O_k)$.

A forward-backward algo is used for finding probable state paths.

Forward-Backward Algorithm

Forward Probability

- **$\alpha_k(i)$ = F(k, i) = Probability of being in state S_i after having seen $O_0O_1O_2$ …. O_k**

i.e., The forward variable **$\alpha_k(i)$ or F(k, i)** is defined as the joint probability of the partial observation sequence $O_0\,O_1\,O_2 \ldots\, O_k$ and that the hidden state at time k is S_i :

- **$\alpha_k(i)$ = F(k, i) = $P(O_0O_1O_2 \ldots O_k, S_i)$**
- With m as length of observation sequence

P(Observed Sequence)= **$P(O_0O_1O_2 \ldots O_m)$**

$$= \sum_{i=0}^{N} P(O_0 O_1 O_2 \ldots O_m , S_i)$$

Forward Probability

- P(Observed Sequence)= $\sum_{i=0}^{N} F(m,i)$

- Method to re-compute forward recursively,

F(k, q) = $\sum_{p=0}^{N} F(k-1,p)\, P(S_p \xrightarrow{O_k} S_q)$

Backward Probability

- We define backward probability as:

B(k, i) = Probability of seeing $O_k O_{k+1} O_{k+2} \dots O_m$ given that the state was S_i .

$B(k, i) = P(O_k O_{k+1} O_{k+2} \dots O_m / S_i)$

Where m is length of observed sequence.

This is better expressed by working through the mathematical calculation

Backward Probability

$$O_0 \quad O_1 \; .... \; O_{k-1} \quad O_k \quad O_{k+1} \; ..... \; O_{m-1} \quad O_m$$

$$S_0 \rightarrow \qquad S_p \quad S_q \qquad S_m \qquad S_{final}$$

- So, the backward probability B(k, p) is the probability of seeing $O_k O_{k+1} O_{k+2} \ldots O_m$ given that the state was S_p .

$$B(k, p) = P(O_k O_{k+1} O_{k+2} \ldots O_m / S_p)$$

$$B(k, p) = P(O_{k+1} O_{k+2} \ldots O_m , O_k / S_p)$$

perform marginalization

$$B(k, p) = \sum_{q=0}^{N} P(O_{k+1} O_{k+2} \ldots O_m, O_k, S_q / S_p)$$

- So, backward probability is :

$$\textbf{B(k, p)} = \sum_{q=0}^{N} P(O_k, S_q / S_p\ ) . P(O_{k+1} O_{k+2} \ldots O_m / O_k, S_q, S_p\ )$$

$$= \sum_{q=0}^{N} P(O_{k+1} O_{k+2} \ldots O_m / \ S_q) \,.\, P(O_k, S_q / S_p\ )$$

$$= \sum_{q=0}^{N} B(k+1\,,\ q) \,.\, P(S_p \xrightarrow{O_k} S_q\ )$$

where ,

K+1 goes on increasing towards end of observation sequence.

Boundary Condition for B(k, p)

- Boundary condition for B(k, p) is obtained from the last symbol B(m, final) where S_{final} is one of the states of HMM.

Backward Probability

- $B(k, i)$ = Probability of seeing $O_k O_{k+1} O_{k+2} \ldots O_m$ given that the state was S_i.
- $B(k, i) = P(O_k O_{k+1} O_{k+2} \ldots O_m / S_i)$
- With m as length of observation sequence

$$
\begin{aligned}
P(\text{Observed Sequence}) &= P(O_0 O_1 O_2 \ldots O_m) \\
&= P(O_0 O_1 O_2 \ldots O_m / S_0) \\
&= B(0,0)
\end{aligned}
$$

$$B(k, p) = \sum_{q=0}^{N} B(k+1, q) \,.\, P(S_p \xrightarrow{O_k} S_q)$$

Baum Welch algorithm

- ✓ Training Hidden Markov Model (not structure learning, i.e., the structure of the HMM is pre-given). This involves:
- ✓ Learning probability values ONLY

The Learning Problem

Learning: Given an observation sequence O and the set of possible states in the HMM, learn the HMM parameters A and B.

- **Baum-Welch** = **Forward-Backward Algorithm** (Baum 1972)
- Is a special case of the EM or Expectation-Maximization algorithm (Dempster, Laird, Rubin)
- The algorithm will let us train the transition probabilities $A = \{a_{ij}\}$ and the emission probabilities $B = \{b_i(o_t)\}$ of the HMM

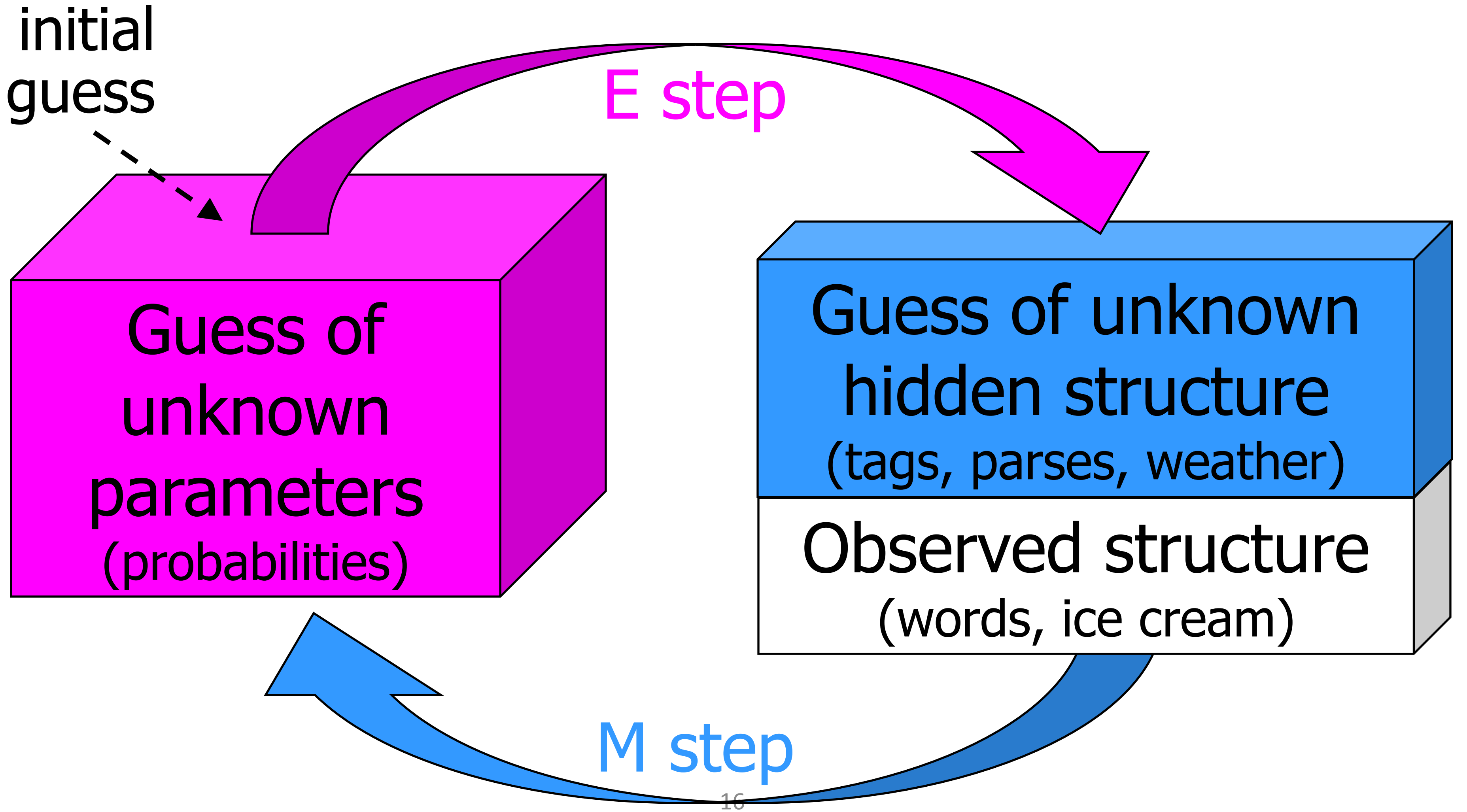
General Idea

- Start by devising a model
  - Any model that predicts the corpus observations via some hidden structure (tags, …)
- Initially **guess** the parameters of the model!
  - Educated guess is best, but random can work

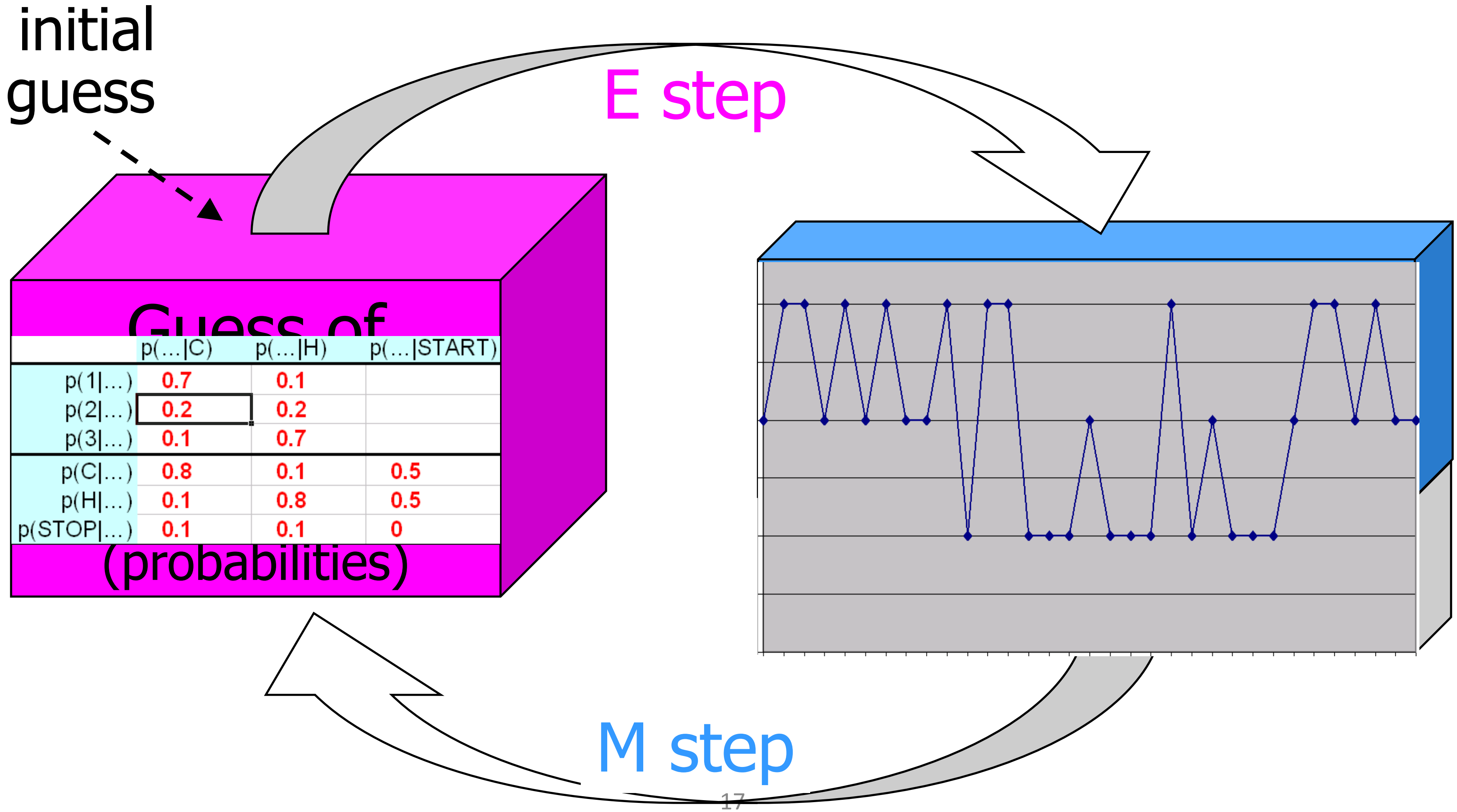
- **Expectation step:** Use current parameters (and observations) to reconstruct hidden structure
- **Maximization step:** Use that hidden structure (and observations) to re-estimate parameters

Repeat until convergence!

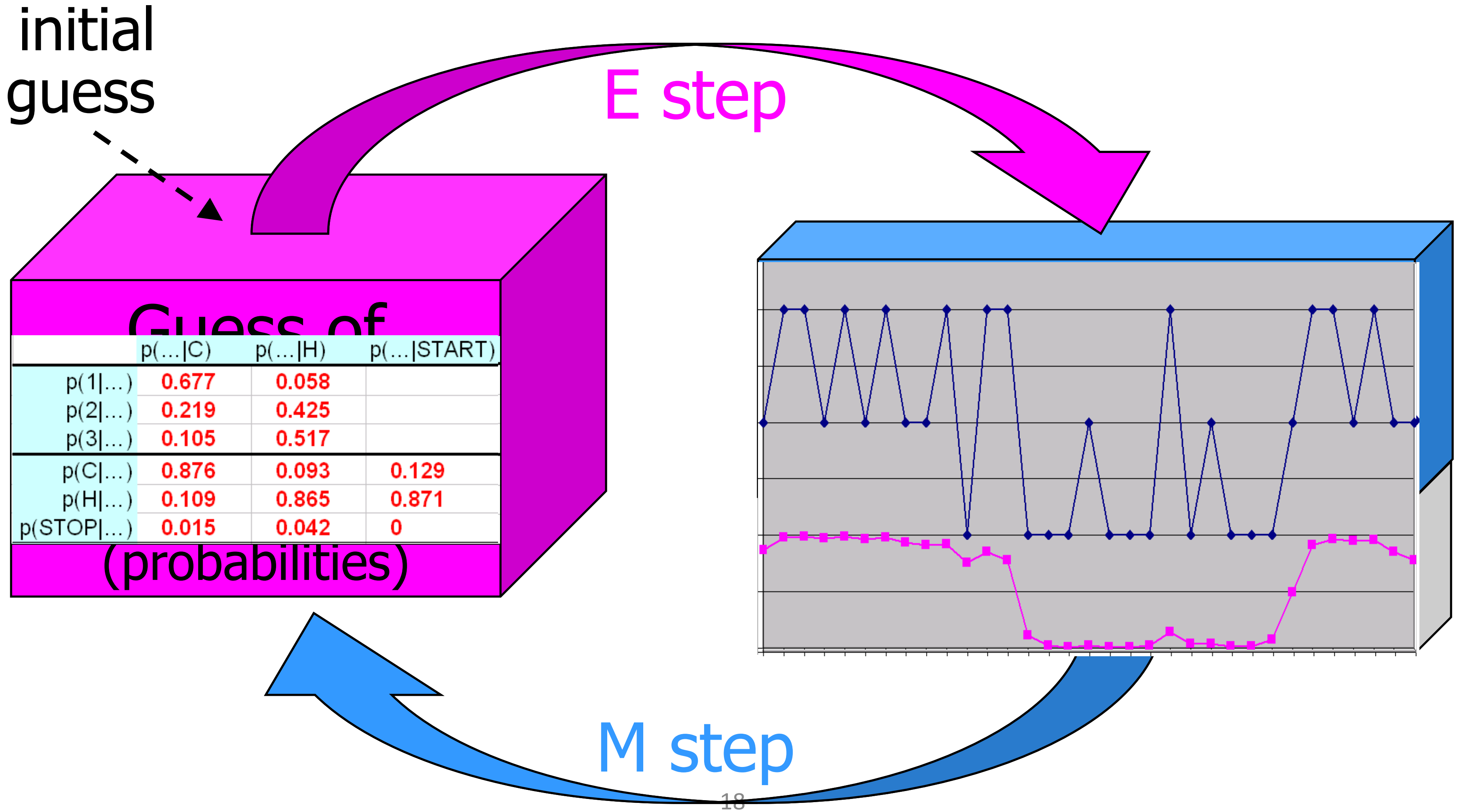
General Idea



For Hidden Markov Models

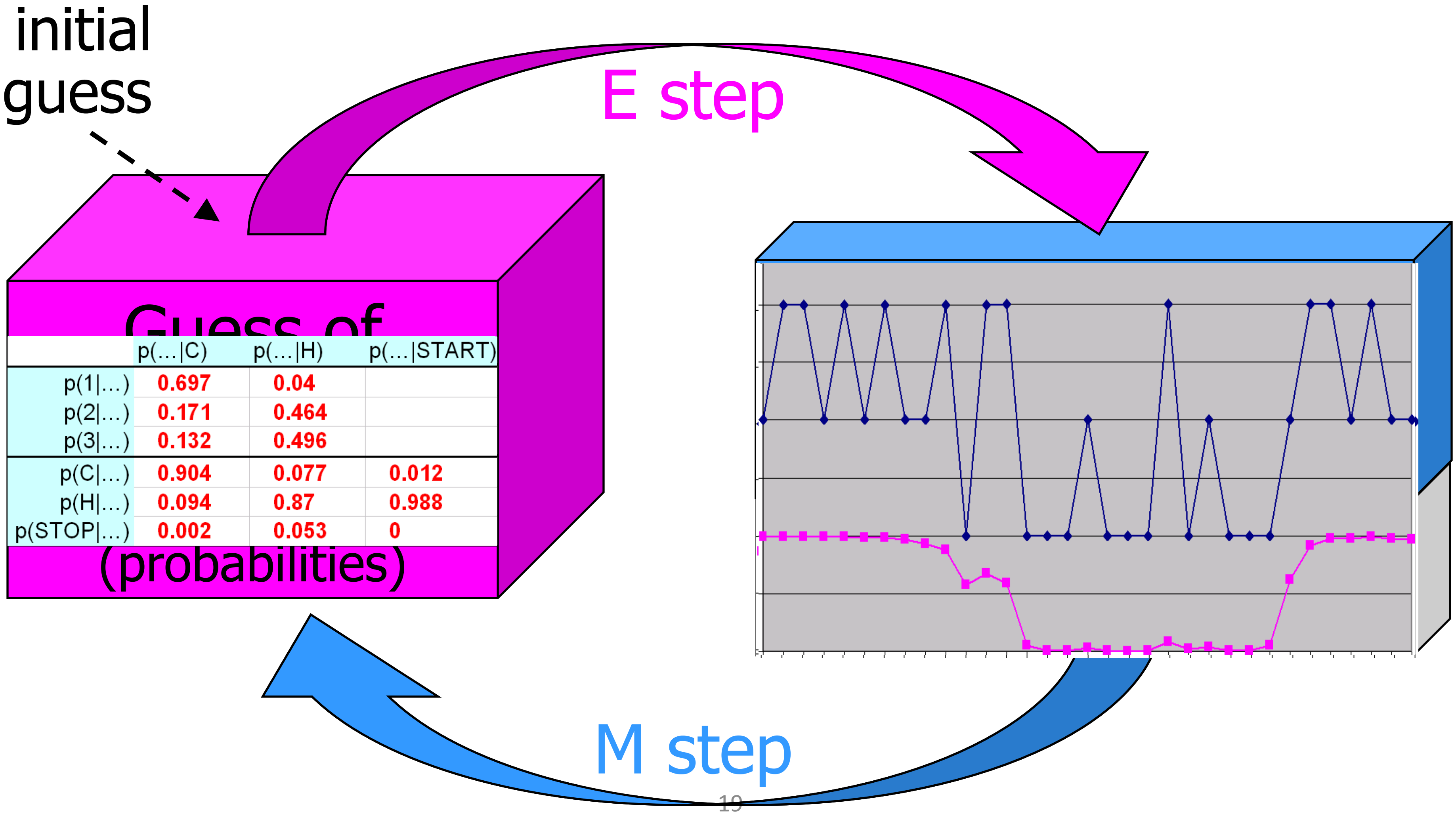


For Hidden Markov Models



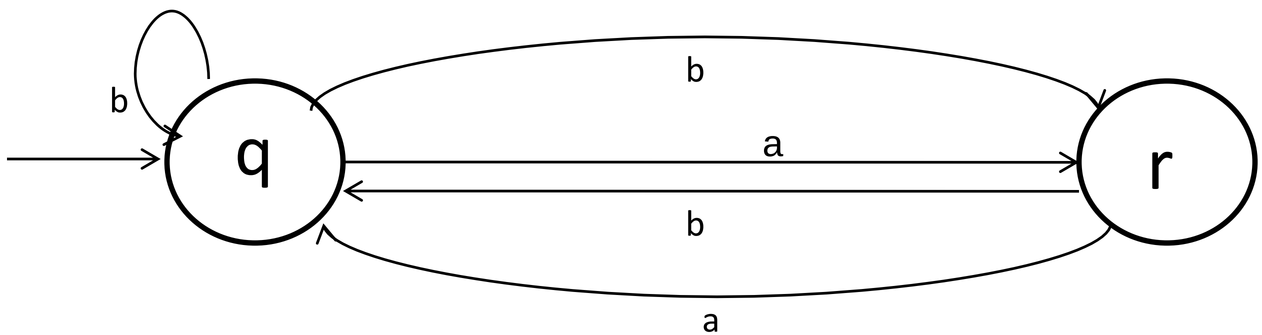
	p(...|C)	p(...|H)	p(...|START)
p(1|...)	0.677	0.058	
p(2|...)	0.219	0.425	
p(3|...)	0.105	0.517	
p(C|...)	0.876	0.093	0.129
p(H|...)	0.109	0.865	0.871
p(STOP|...)	0.015	0.042	0

For Hidden Markov Models



	p(…|C)	p(…|H)	p(…|START)
p(1|…)	0.697	0.04	
p(2|…)	0.171	0.464	
p(3|…)	0.132	0.496	
p(C|…)	0.904	0.077	0.012
p(H|…)	0.094	0.87	0.988
p(STOP|…)	0.002	0.053	0

Start of baum-welch algorithm



Training sequence/Observation sequence = abb aaa bbb aaa

Sequence of states with respect to input symbols

o/p seq

$$\rightarrow q \xrightarrow{a} r \xrightarrow{b} q \xrightarrow{b} q \xrightarrow{a} r \xrightarrow{a} q \xrightarrow{a} r \xrightarrow{b} q \xrightarrow{b} q \xrightarrow{b} q \xrightarrow{a} r \xrightarrow{a} q \xrightarrow{a} r$$

State seq

Calculating probabilities from table

$$P(q \xrightarrow{a} r) = 5/8$$

$$P(q \xrightarrow{b} q) = 3/8$$

$$P(s^i \xrightarrow{w_k} s^j) = \frac{c(s^i \xrightarrow{w_k} s^j)}{\sum_{l=1}^{T} \sum_{m=1}^{A} c(s^i \xrightarrow{w_m} s^l)}$$

T=#states

A=#alphabet symbols

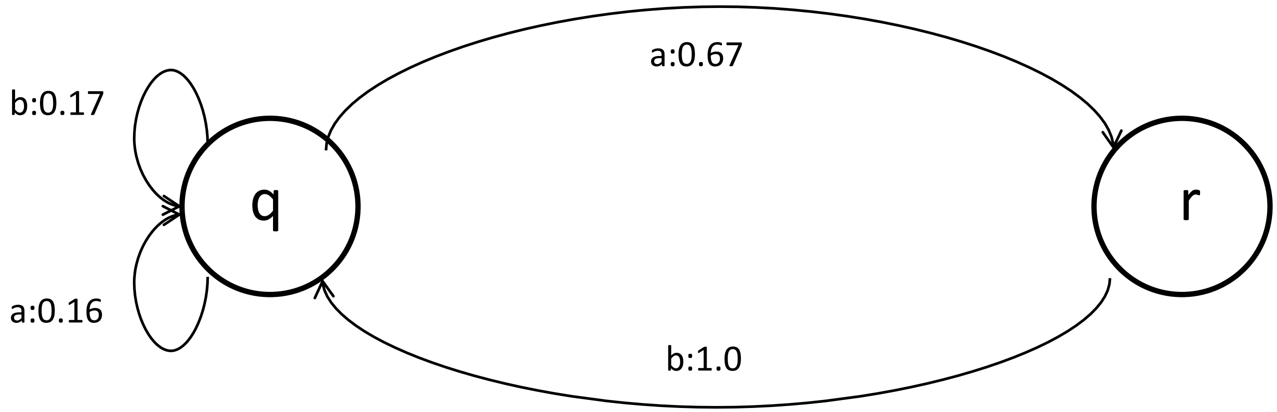
Table of counts

Src	Dest	O/P	Count
q	r	a	5
q	q	b	3
r	q	a	3
r	q	b	2

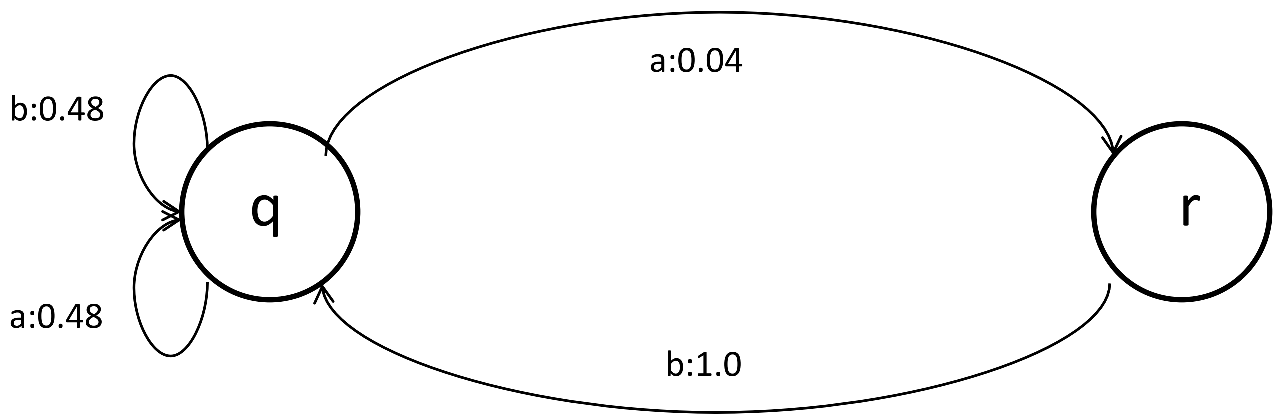
Now if we have a non-deterministic transitions then multiple state seq possible for the given o/p seq (ref. to previous slide's feature). Our aim is to find expected count through this.

Learning probabilities

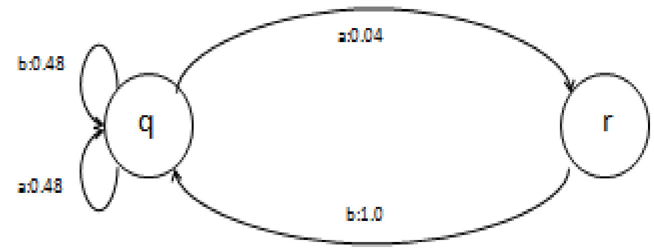
Machine Given : Learn from it



Initial guess



One run of Baum-Welch algorithm: *string* ***ababb***



$\in \to a$	$a \to b$	$b \to a$	$a \to b$	$b \to b$	$b \to \in$	P(path)	$q \xrightarrow{a} r$	$r \xrightarrow{b} q$	$q \xrightarrow{a} q$	$q \xrightarrow{b} q$
q	r	q	r	q	q	0.00077				
q	r	q	q	q	q					
q	q	q	r	q	q					
q	q	q	q	q	q					
Expected Count →										
New Probabilities (P) →										

State sequences

* $\in$ is considered as starting and ending symbol of the input sequence string

P(state seq:qrqrqq)=

$$P(qrqrqq) = P(S_0 \xrightarrow{\in} q) * P(q \xrightarrow{a} r) * P(r \xrightarrow{b} q) * P(q \xrightarrow{a} r) * P(r \xrightarrow{b} q) * P(q \xrightarrow{b} q)$$

$$= 1 * 0.04 * 1.0 * 0.04 * 1.0 * 0.48$$

$$= 0.00077$$

One run of Baum-Welch algorithm: *string **ababb***

$\in \to a$	$a \to b$	$b \to a$	$a \to b$	$b \to b$	$b \to \in$	P(path)	$q \xrightarrow{a} r$	$r \xrightarrow{b} q$	$q \xrightarrow{a} q$	$q \xrightarrow{b} q$
q	r	q	r	q	q	0.00077				
q	r	q	q	q	q	0.00442				
q	q	q	r	q	q	0.00442				
q	q	q	q	q	q	0.02548				
Expected Count →										
New Probabilities (P) →										

State sequences

* $\in$ is considered as starting and ending symbol of the input sequence string

P(qrqrqq) = $P(S_0 \xrightarrow{\in} q) * P(q \xrightarrow{a} r) * P(r \xrightarrow{b} q) * P(q \xrightarrow{a} r) * P(r \xrightarrow{b} q) * P(q \xrightarrow{b} q)$

= 1* 0.04 * 1.0 *0.04 * 1.0 *0.48

= 0.00077

One run of Baum-Welch algorithm: *string* ***ababb***

$\in \rightarrow a$	$a \rightarrow b$	$b \rightarrow a$	$a \rightarrow b$	$b \rightarrow b$	$b \rightarrow \in$	P(path)	$q \xrightarrow{a} r$	$r \xrightarrow{b} q$	$q \xrightarrow{a} q$	$q \xrightarrow{b} q$
q	r	q	r	q	q	0.00077				
q	r	q	q	q	q	0.00442				
q	q	q	r	q	q	0.00442				
q	q	q	q	q	q	0.02548				
Expected Count →						**0.035**				
New Probabilities (P) →										

State sequences

* $\in$ is considered as starting and ending symbol of the input sequence string

Expected Count (path) = Summation(path Column)

= (0.00077+0.00442+0.00442+0.2548)

= 0.03509 ≈ **0.035**

One run of Baum-Welch algorithm: *string* ***ababb***

$\in \rightarrow a$	$a \rightarrow b$	$b \rightarrow a$	$a \rightarrow b$	$b \rightarrow b$	$b \rightarrow \in$	P(path)	$q \xrightarrow{a} r$	$r \xrightarrow{b} q$	$q \xrightarrow{a} q$	$q \xrightarrow{b} q$
q	r	q	r	q	q	0.00077	0.00154			
q	r	q	q	q	q	0.00442				
q	q	q	r	q	q	0.00442				
q	q	q	q	q	q	0.02548				
Expected Count →						0.035				
New Probabilities (P) →										

Count for state transitions, i.e., $q \xrightarrow{a} r$, $r \xrightarrow{b} q$, and so on.

Count($q \xrightarrow{a} r$) = path * number of transitions = 0.00077 * 2 = 0.00154

One run of Baum-Welch algorithm: *string ababb*

$\in \rightarrow a$	$a \rightarrow b$	$b \rightarrow a$	$a \rightarrow b$	$b \rightarrow b$	$b \rightarrow \in$	P(path)	$q \xrightarrow{a} r$	$r \xrightarrow{b} q$	$q \xrightarrow{a} q$	$q \xrightarrow{b} q$
q	r	q	r	q	q	0.00077	0.00154	0.00154	0	0.00077
q	r	q	q	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	r	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	q	q	q	0.02548	0.0	0.000	0.05096	0.07644
Expected Count →						0.035	0.01	0.01	0.06	0.095
New Probabilities (P) →										

One run of Baum-Welch algorithm: *string* ***ababb***

$\in \to a$	$a \to b$	$b \to a$	$a \to b$	$b \to b$	$b \to \in$	P(path)	$q \xrightarrow{a} r$	$r \xrightarrow{b} q$	$q \xrightarrow{a} q$	$q \xrightarrow{b} q$
q	r	q	r	q	q	0.00077	0.00154	0.00154	0	0.00077
q	r	q	q	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	r	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	q	q	q	0.02548	0.0	0.000	0.05096	0.07644
Expected Count →						0.035	0.01	0.01	0.06	0.095
New Probabilities (P) →							0.06			

Estimating new probabilities for state sequences, i.e., $q \xrightarrow{a} r$, $r \xrightarrow{b} q$, and so on.

Normalized new probability for $q \xrightarrow{a} r$ $= \dfrac{\text{expected count of } q \xrightarrow{a} r}{\sum \text{expected count of } q \xrightarrow{*} *}$

$$= \frac{0.01}{0.01 + 0.06 + 0.095}$$

= 0.06

One run of Baum-Welch algorithm: *string ababb*

$\in \to a$	$a \to b$	$b \to a$	$a \to b$	$b \to b$	$b \to \in$	P(path)	$q \xrightarrow{a} r$	$r \xrightarrow{b} q$	$q \xrightarrow{a} q$	$q \xrightarrow{b} q$
q	r	q	r	q	q	0.00077	0.00154	0.00154	0	0.00077
q	r	q	q	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	r	q	q	0.00442	0.00442	0.00442	0.00442	0.00884
q	q	q	q	q	q	0.02548	0.0	0.000	0.05096	0.07644
Expected Count →						0.035	0.01	0.01	0.06	0.095
New Probabilities (P) →							0.06	1.0	0.36	0.581

State sequences

Note:
- Now new path values can be recomputed.
- So new expected counts which will lead to new probability values.
- This way through multiple iterations the probability values will converge.

What is an epoch?

- It is one iteration over all observation patterns.
- So, after each iteration we get updated probabilities. This is done until converges.

The Baum-Welch Algorithm

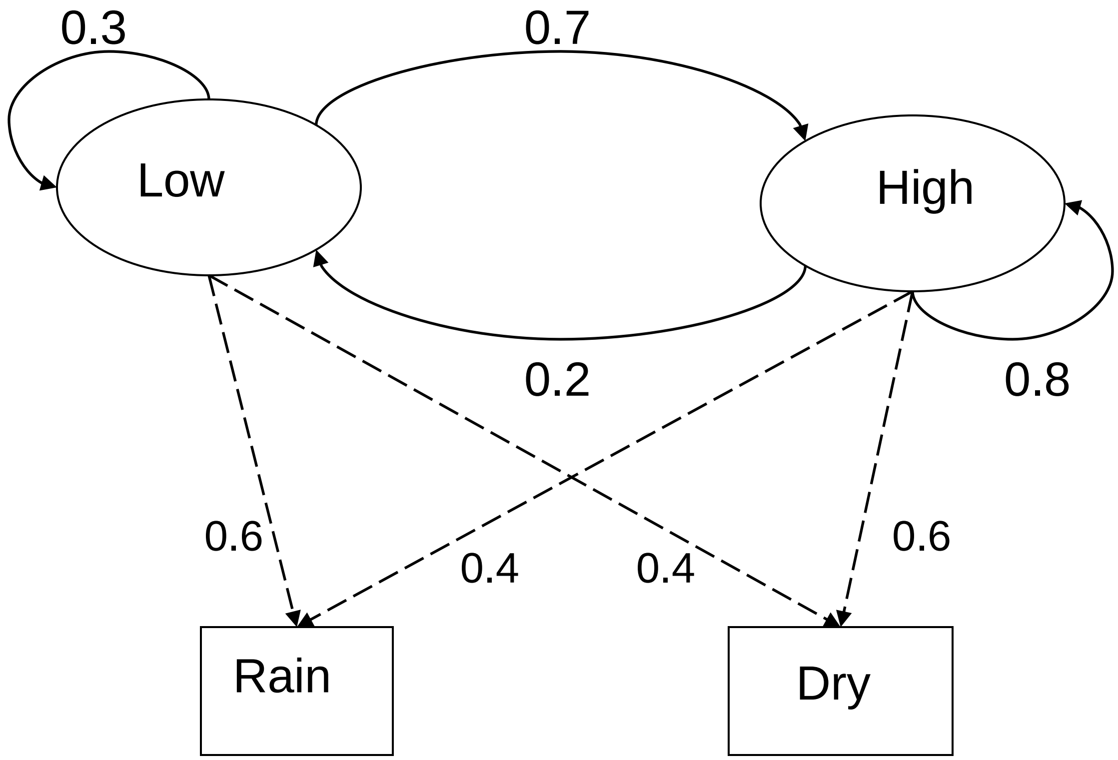
- initialize the parameters of the HMM
- iterate until convergence
  - initialize $n_{k,c}$, $n_{k \to l}$ with pseudocounts
  - **E-step**: for each training set sequence *j* = 1…*n*
    - calculate $f_k(i)$ values for sequence *j*
    - calculate $b_k(i)$ values for sequence *j*
    - add the contribution of sequence *j* to $n_{k,c}$, $n_{k \to l}$
  - **M-step**: update the HMM parameters using $n_{k,c}$, $n_{k \to l}$

where,

$n_{k,c}$ be the expected number of emissions of *c* from state *k* for the training set, and

$n_{k \to l}$ be the expected number of transitions from state *k* to state *l* for the training set

Example of Hidden Markov Model



Example of Hidden Markov Model

1. Two states : ‘Low’ and ‘High’ atmospheric pressure.
2. Two observations : ‘Rain’ and ‘Dry’.
3. Transition probabilities: P(‘Low’|‘Low’)=0.3 , P(‘High’|‘Low’)=0.7 , P(‘Low’|‘High’)=0.2, P(‘High’|‘High’)=0.8
4. Observation probabilities : P(‘Rain’|‘Low’)=0.6 , P(‘Dry’|‘Low’)=0.4 , P(‘Rain’|‘High’)=0.4 , P(‘Dry’|‘High’)=0.3 .
5. Initial probabilities: say P(‘Low’)=0.4 , P(‘High’)=0.6 .

Calculation of observation sequence probability

1. Suppose we want to calculate a probability of a sequence of observations in our example, {'Dry','Rain'}.
2. Consider all possible hidden state sequences:
3. P({'Dry','Rain'}) = P({'Dry','Rain'} , {'Low','Low'}) + P({'Dry','Rain'} , {'Low','High'}) + P({'Dry','Rain'} , {'High','Low'}) + P({'Dry','Rain'} , {'High','High'})

where first term is :

P({'Dry','Rain'} , {'Low','Low'})= P({'Dry','Rain'} | {'Low','Low'}) P({'Low','Low'}) = P('Dry'|'Low')P('Rain'|'Low') P('Low')P('Low'|'Low)

= 0.4*0.4*0.6*0.4*0.3