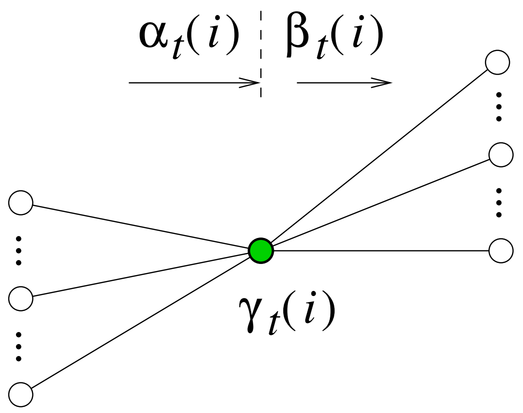
HMM Training

Dr Philip Jackson

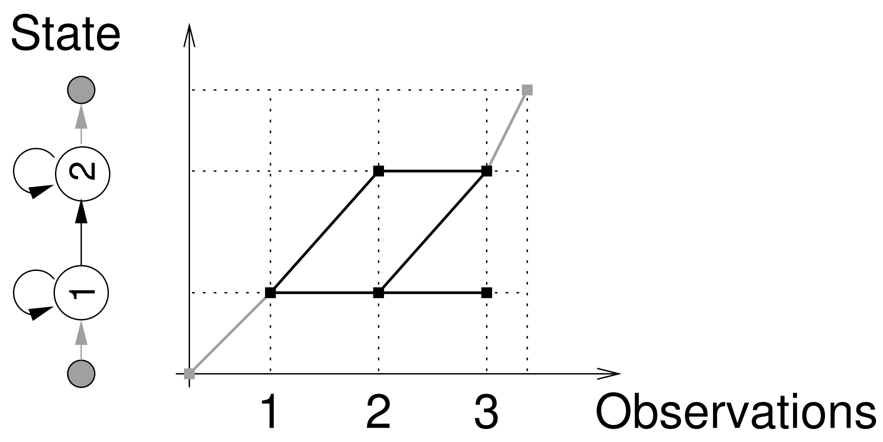
- Task 3: Training the models
  - Viterbi re-estimation
  - Expectation maximisation
  - Occupation & transition likelihoods
  - Baum-Welch formulae



HMM Recognition & Training

Three tasks within HMM framework

1. Compute likelihood of a set of observations for a given model, $P(\mathcal{O}|\lambda)$
2. Decode a test sequence by calculating the most likely path, X^*
3. Optimise pattern templates by training the model parameters, $\Lambda = \{\lambda\}$



Task 1: Forward procedure

To calculate **forward likelihood**, $\alpha_t(i) = P(\mathbf{o}_1^t, x_t = i|\lambda)$:

1. Initialise at $t = 1$,

   $$\alpha_1(i) = \pi_i \, b_i(o_1) \qquad \text{for } 1 \le i \le N$$

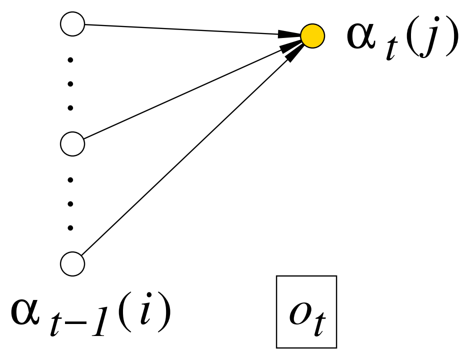
2. Recur for $t = \{2, 3, \ldots, T\}$,

   $$\alpha_t(j) = \left[\sum_{i=1}^{N} \alpha_{t-1}(i) \, a_{ij}\right] b_j(o_t) \qquad \text{for } 1 \le j \le N \tag{1}$$

3. Finalise,

   $$P(\mathcal{O}|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \, \eta_i$$

Thus Task 1 is solved efficiently by recursion.



Task 1: Backward procedure

We define **backward likelihood**, $\beta_t(i) = P(\mathbf{o}_{t+1}^T | x_t = i, \lambda)$, and calculate:

1. Initialise at $t = T$,

   $\beta_T(i) = \eta_i$ for $1 \leq i \leq N$

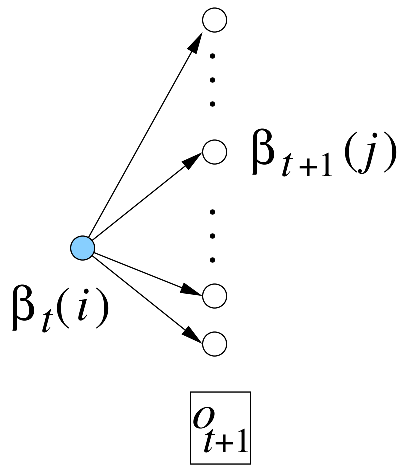
2. Recur for $t = \{T-1, T-2, \ldots, 1\}$,

   $\beta_t(i) = \sum_{j=1}^{N} a_{ij}\, b_j(o_{t+1})\, \beta_{t+1}(j)$ for $1 \leq i \leq N$ (2)

3. Finalise,

   $P(\mathcal{O}|\lambda) = \sum_{i=1}^{N} \pi_i\, b_i(o_1)\, \beta_1(i)$

This equivalently computes $P(\mathcal{O}|\lambda)$.



Task 2: Viterbi decoding of the best path

To compute the **maximum cumulative likelihood**, $\delta_t(i)$:

1. Initialise at $t = 1$,
   $\delta_1(i) = \pi_i b_i(o_1)$
   $\psi_1(i) = 0$ for $1 \le i \le N$

2. Recur for $t = \{2, 3, \dots, T\}$,
   $\delta_t(j) = \max_i \left[\delta_{t-1}(i) a_{ij}\right] b_j(o_t)$
   $\psi_t(j) = \arg\max_i \left[\delta_{t-1}(i) a_{ij}\right]$ for $1 \le j \le N$

3. Finalise,
   $P(\mathcal{O}, X^* | \lambda) = \max_i [\delta_T(i)\eta_i]$
   $x_T^* = \arg\max_i [\delta_T(i)\eta_i]$

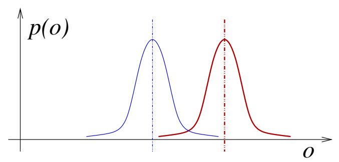
4. Trace back, for $t = \{T, T-1, \dots, 2\}$,

$$x_{t-1}^* = \psi_t\left(x_t^*\right), \quad \text{and} \quad X^* = \{x_1^*, x_2^*, \dots, x_T^*\} \tag{3}$$

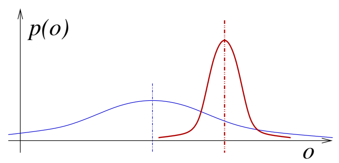
Task 3: training the models

Techniques for optimal parameter estimation

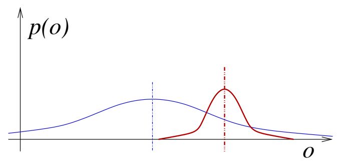
- Least-squares (LS) estimation
  - based on minimum mean squared error
  - all classes' errors treated equally



- **Maximum likelihood (ML) estimation**
  - based on likelihood
  - consider distribution within a class

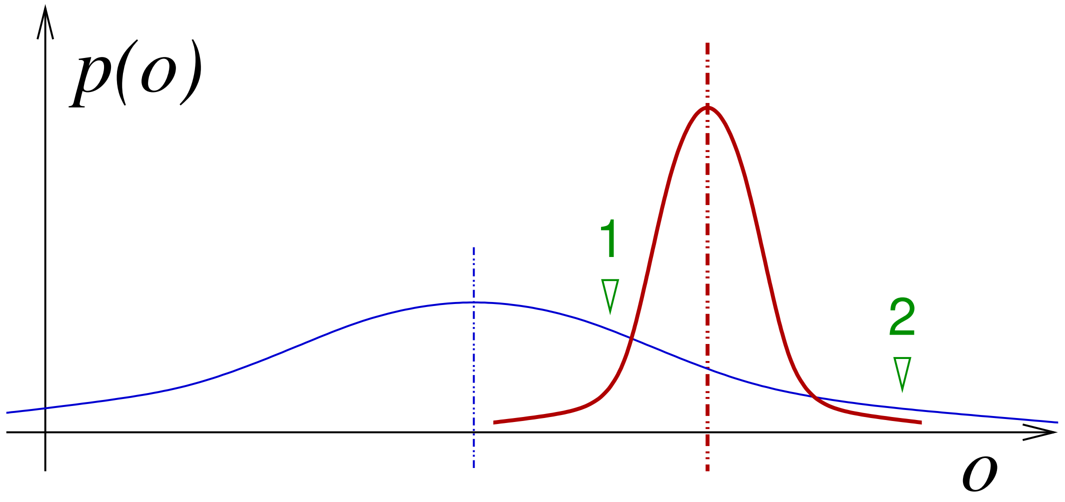


- Maximum a posteriori (MAP) estimation
  - based on posterior probability
  - give weights across the classes



Motivation for the most likely model parameters

Given two different probability density functions (pdfs),



how would you classify observations in regions 1 and 2 by least-squares, and compare this to their relative likelihoods?

Maximum likelihood training

In general, we seek the value of some model parameter c that is most likely to yield our set of training data, $\mathcal{O}_{\text{train}}$

The maximum likelihood (ML) estimate $\hat{c}$ is found setting the derivative of $P(\mathcal{O}_{\text{train}}|c)$ w.r.t. c equal to zero:

$$\frac{\partial \ln P(\mathcal{O}_{\text{train}}|\hat{c})}{\partial c} = 0 \tag{4}$$

Solving the **likelihood equation** tells us how to optimise the model parameters in training.

According to different assumptions, we will examine:

- Viterbi training
- Baum-Welch training

Re-estimating the parameters of the model λ

As a preliminary approach, use the optimal path X^* computed by the Viterbi algorithm with initial model parameters $\lambda = \{A, B\}$. In so doing, we approximate the total likelihood of the observations:

$$\begin{aligned} P(\mathcal{O}|\lambda) &= \sum_X P(\mathcal{O}, X|\lambda) \\ &\approx \max_X P(\mathcal{O}, X|\lambda) \\ &= P(\mathcal{O}, X^*|\lambda) \qquad (5) \end{aligned}$$

Using X^*, we make a hard binary decision about the state occupation, $q_t(i) \in \{0, 1\}$, and train the parameters of our model accordingly.

Viterbi training (hard state assignment)

Model parameters can be re-estimated using the Viterbi alignment to assign observations to states (a.k.a. segmental k-means training).

State-transition probabilities,

$$\hat{a}_{ij} = \frac{\sum_{t=2}^{T} q_{t-1}(i)\, q_t(j)}{\sum_{t=1}^{T} q_t(i)} \qquad \text{for } 1 \leq i, j \leq N$$

where state indicator $q_t(i) = \begin{cases} 1 & \text{for } i = x_t \\ 0 & \text{otherwise} \end{cases}$

Discrete output probabilities,

$$\hat{b}_j(k) = \frac{\sum_{t=1}^{T} q_t(j)\, \omega_t(k)}{\sum_{t=1}^{T} q_t(j)} \qquad \text{for } 1 \leq j \leq N \text{ and } 1 \leq k \leq K$$

where event indicator $\omega_t(k) = \begin{cases} 1 & \text{for } k = o_t \\ 0 & \text{otherwise} \end{cases}$

Viterbi re-estimation (multiple files)

Generally, we use a set of training sequences, $r \in \{1, \ldots, R\}$:

(a) State-transition probabilities,

$$\hat{a}_{ij} = \frac{\sum_{r=1}^{R} \sum_{t=2}^{T} q_{t-1}^{r}(i) q_t^r(j)}{\sum_{r=1}^{R} \sum_{t=1}^{T} q_t^r(i)} \qquad \text{for } 1 \le i, j \le N$$

(b) Discrete output probabilities,

$$\hat{b}_j(k) = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T} q_t^r(j)\, \omega_t^r(k)}{\sum_{r=1}^{R} \sum_{t=1}^{T} q_t^r(j)} \qquad \text{for } 1 \le j \le N \text{ and } 1 \le k \le K$$

Together these new parameters make up our re-estimated model $\hat{\lambda} = \{\hat{A}, \hat{B}\}$.

Worked example of Viterbi re-estimation

state indicators

$q_1(1) =$

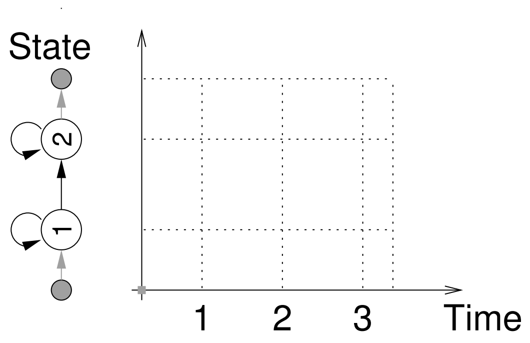
$q_1(2) =$

$q_2(1) =$

$q_2(2) =$

$q_3(1) =$

$q_3(2) =$



$\mathcal{O} = \{ \qquad \}$

$\hat{A} =$

$\hat{B} =$

Maximum likelihood training by EM

Baum-Welch re-estimation (occupation)

Yet, the hidden state occupation is not known with absolute certainty. The expectation maximisation (EM) technique optimises the model parameters based on soft assignment of observations to states via the **occupation likelihood**,

$$\gamma_t(i) = P\left(x_t = i | \mathcal{O}, \lambda\right) \tag{6}$$

relaxing the assumption previously made in eq. 5.

Using Bayes theorem, and rearranging, gives

$$\begin{aligned} \gamma_t(i) &= \frac{P\left(\mathcal{O}, x_t = i | \lambda\right)}{P(\mathcal{O}|\lambda)} \\ &= \frac{\alpha_t(i)\,\beta_t(i)}{P(\mathcal{O}|\lambda)} \end{aligned} \tag{7}$$

where α_t, β_t and $P(\mathcal{O}|\lambda)$ are computed by the forward and backward procedures, which also give $P(\mathcal{O}|\lambda)$.

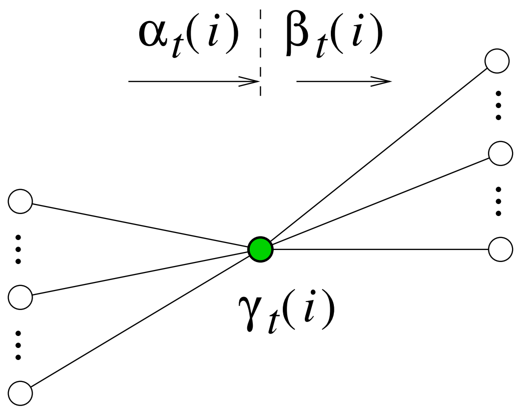
Baum-Welch re-estimation (transition)

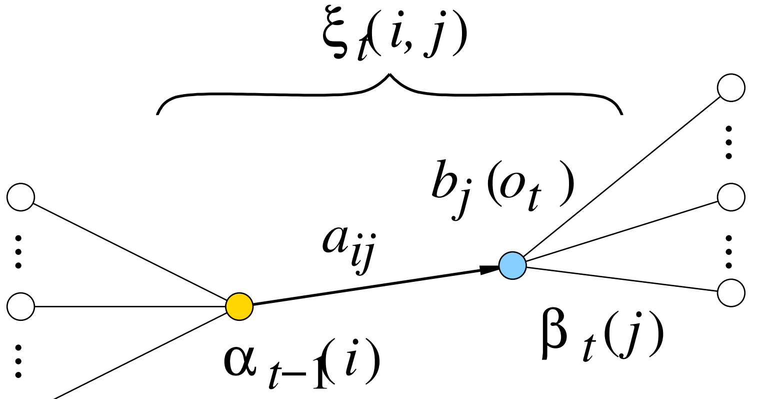
Similarly, we define the **transition likelihood**,

$$\xi_t(i,j) = P\left(x_{t-1}=i, x_t=j|\mathcal{O},\lambda\right) = \frac{P\left(\mathcal{O}, x_{t-1}=i, x_t=j|\lambda\right)}{P(\mathcal{O}|\lambda)}$$

$$= \frac{P\left(\mathbf{o}_1^{t-1}, x_{t-1}=i|\lambda\right)\, P\left(o_t, x_t=j|x_{t-1}=i,\lambda\right)\, P\left(\mathbf{o}_{t+1}^T|x_t=j,\lambda\right)}{P(\mathcal{O}|\lambda)}$$

$$= \frac{\alpha_{t-1}(i)\ a_{ij}\, b_j(o_t)\ \beta_t(j)}{P(\mathcal{O}|\lambda)} \tag{8}$$





Trellis depiction of (left) occupation and (right) transition likelihoods

Baum-Welch training (soft state assignment)

State-transition probabilities,

$$\hat{a}_{ij} = \frac{\sum_{t=2}^{T} \xi_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)} \qquad \text{for } 1 \leq i, j \leq N$$

Discrete output probabilities,

$$\hat{b}_j(k) = \frac{\sum_{t=1}^{T} \gamma_t(j)\, \omega_t(k)}{\sum_{t=1}^{T} \gamma_t(j)} \qquad \text{for } 1 \leq j \leq N \text{ and } 1 \leq k \leq K$$

Re-estimation increases the likelihood over the training data with the new model $\hat{\lambda}$ until it converges at a local optimum,

$$P(\mathcal{O}_{\text{train}}|\hat{\lambda}) \geq P(\mathcal{O}_{\text{train}}|\lambda)$$

although it does not guarantee a global maximum.

Worked example of Baum-Welch re-estimation

occupation likelihoods

$\gamma_1(1) =$

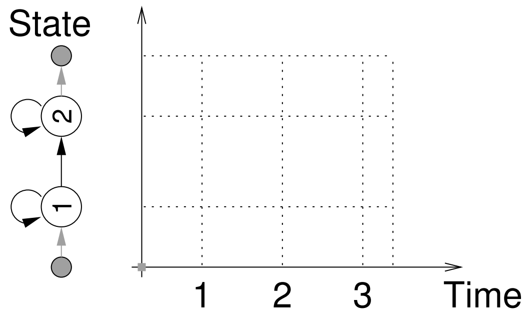
$\gamma_1(2) =$

$\gamma_2(1) =$

$\gamma_2(2) =$

$\gamma_3(1) =$

$\gamma_3(2) =$



$\mathcal{O} = \{ \qquad\qquad \}$

transition likelihoods

$\xi_1(1,1) =$

$\xi_1(2,1) =$

$\xi_2(1,1) =$

$\xi_2(2,1) =$

$\xi_3(1,1) =$

$\xi_3(2,1) =$

$\xi_1(1,2) =$

$\xi_1(2,2) =$

$\xi_2(1,2) =$

$\xi_2(2,2) =$

$\xi_3(1,2) =$

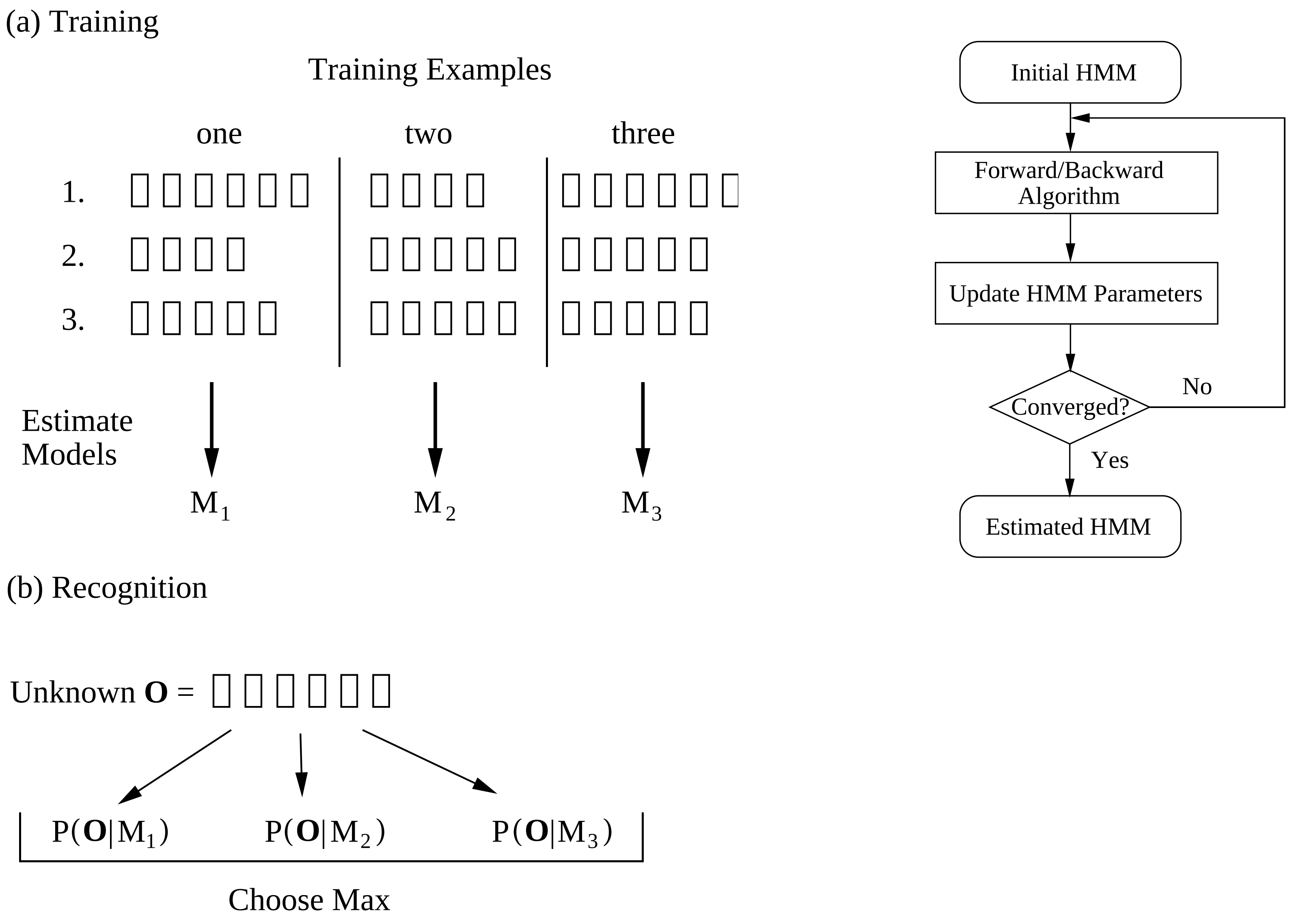
$\xi_3(2,2) =$

Worked example (continued)

$$\hat{A} =$$

$$\hat{B} =$$

Use of HMMs with training and test data



Isolated word training and recognition (Young et al., 2002)

Part 3 summary

- Task 1: Forward/backward likelihoods, α_t and β_t
- Task 2: Viterbi decoding, Q^* and X^*
- Task 3: Training model parameters, $\lambda = \{A, B\}$
  - Viterbi re-estimation (hard assignment)
  - Baum-Welch re-estimation (soft assignment) using occupation and transition likelihoods, γ_t and ξ_t
- Practical training procedure

Homework

Complete the worked examples:

1. Viterbi training with the model you used last week and the state sequence $X = \{1, 2, 2\}$
   - re-estimate the π_i and a_{ij} parameters
   - re-estimate $b_i(k)$ for one state i considering red (k=1), green (k=2) and blue (k=3) observations
2. Repeat for Baum-Welch training using the α_t and β_t values from last time, and compare the results.