Topic 4

Expectation-Maximization

4.1 Identification of a Gaussian mixture

Common sense seems to tell us that the lesser the number of unknown variables, the simpler the optimization problem. Contrary to this conventional opinion, Expectation-Maximization (EM) techniques simplify a solution of difficult maximum likelihood problems by adding more unobserved, or latent variables. This enlargement of the space of unknowns with latent variables is called data augmentation.

To illustrate this approach, let us consider a simple example of Gaussian mixture identification by searching for the MLE of its parameters\(^1\). The goal is to find the mixture

\[
p(x|\theta) = \sum_{k=1}^{K} \alpha_k \phi(x|\theta_k)
\]

(4.1)

such that closely approximates a given collection of actual measurements \(\mathbf{x} = (x_1, \ldots, x_n)\). Here, \(\theta = \{ (\alpha_k, \theta_k) : k = 1, \ldots, K \} \) denotes a set of the mixture parameters, \(\alpha_k\) is a prior probability of the \(k\)-th Gaussian \(\phi(x|\theta_k)\) in the mixture, and \(\theta_k \equiv (\mu_k, \sigma_k^2)\) denotes the mean and the variance of the \(k\)-th Gaussian.

The log-likelihood of \(n\) i.i.d. sample measurements \(\mathbf{x}\) from the mixed density of

---

\(^1\)Although a general EM framework was introduced first by Dempster, Laird, and Rubin [5] in 1977, the EM algorithm for identification of arbitrary mixture densities (not necessarily the Gaussian mixtures) appeared a decade earlier, in 1968, under the name of unsupervised learning algorithm of Schlesinger [19].
Eq. (4.1) is

$$
\ell(\theta; x) = \ln \prod_{i=1}^{n} p(x_i|\theta) \equiv \sum_{i=1}^{n} \ln \left[ \sum_{k=1}^{K} \alpha_k \varphi(x|\theta_k) \right]
$$

Direct search for the MLE of the parameters $\theta$ is very difficult because the sum of the mixture terms is inside the logarithm.

The estimation problem is greatly simplified by involving $n$ latent variables $\gamma = (\gamma_i : i = 1, \ldots, n)$ indicating the Gaussian component each measurement $x_i$ comes from, i.e. $\gamma_i = k$ if $x_i$ comes from the $k$-th Gaussian. If the indicators $\gamma$ had been known, the log-likelihood would have become very simple:

$$
\ell(\theta; x, \gamma) = \sum_{i=1}^{n} \ln (\alpha_{\gamma_i} \varphi(x_i|\theta_{\gamma_i})) \tag{4.2}
$$

Then the MLE for $\theta_k$ are merely the sample mean and the variance of the subset of measurements $x_k = \{x_i : x_i \in x; \gamma_i = k\}$ sampled from the $k$-th Gaussian, and the MLE for $\alpha_k$ is just the relative size of this subset: $|x_k| = \frac{1}{n} \sum_{i=1}^{n} \delta(\gamma_i - k)$ where $\delta(\ldots)$ is the Kronecker function: $\delta(0) = 1$ and $\delta(z) = 0$ for $z \neq 0$.

Actually, the indicators $\gamma$ are unknown. The EM process is built on the assumption they are random variables such that their probability distribution depends on measurements and mixture parameters. The following two steps are iterated until convergence to the MLEs of the mixture parameters:

| E-step: Each measurement is softly assigned to each component of the mixture using the current weighed MLEs of the mixture parameters. |
| M-step: These soft assignments are used to update the weighed MLEs of the mixture parameters. |

The soft assignment of $x_i$ is performed with conditional probabilities of each mixture component given the parameter values $\theta^\circ = \{ (\alpha_{k,\circ}, \theta_k) : k = 1, \ldots, K \}$. In accord with Bayes’s rule, the conditional probabilities are

$$
p(\gamma_i = k|x_i, \theta^\circ) = \frac{\alpha_{k,\circ} \varphi(x_i|\theta_{k,\circ})}{p(x_i|\theta^\circ)} \equiv \frac{\alpha_{k,\circ} \varphi(x_i|\theta_{k,\circ})}{\sum_{k=1}^{K} \alpha_{k,\circ} \varphi(x_i|\theta_{k,\circ})}
$$
This probability is frequently called the \textit{responsibility} of the \(k\)-th mixture component for the measurement \(x_i\) in order to point to the fact that this probability is obtained for the current estimated, \(\theta^{(t-1)}\), rather than true mixture parameters.

The joint conditional probability of a latent instance \(\gamma = (\gamma_1, \ldots, \gamma_n)\) in the set of all the possible instances \(\mathcal{G}\) of the independently sampled latent indicators is

\[
p(\gamma|x, \theta^o) = \prod_{i=1}^{n} p(\gamma_i|x_i, \theta^o)
\]

It allows for replacing the log-likelihood in Eq. (4.2) being actually a random variable (to be specific, a function of the random latent instances) with its conditional math expectation

\[
Q(\theta, \theta^o) = \sum_{\gamma \in \mathcal{G}} \sum_{i=1}^{n} \ln(\alpha_{\gamma_i} \varphi(x_i|\theta_{\gamma_i})) \prod_{j=1}^{n} p(\gamma_j|x_j, \theta^o)
\]

\[
= \sum_{\gamma_1=1}^{K} \sum_{\gamma_2=1}^{K} \cdots \sum_{\gamma_n=1}^{K} \sum_{i=1}^{n} \ln(\alpha_{\gamma_i} \varphi(x_i|\theta_{\gamma_i})) \prod_{j=1}^{n} p(\gamma_j|x_j, \theta^o)
\]

\[
= \sum_{k=1}^{K} \sum_{i=1}^{n} \ln(\alpha_k \varphi(x_i|\theta_k)) \sum_{\gamma_1=1}^{K} \sum_{\gamma_2=1}^{K} \cdots \sum_{\gamma_n=1}^{K} \delta(\gamma_i - k) \prod_{j=1}^{n} p(\gamma_j|x_j, \theta^o)
\]

\[
= \sum_{k=1}^{K} \sum_{i=1}^{n} \ln(\alpha_k \varphi(x_i|\theta_k)) \ln(p(k|x_i, \theta^o)) + \sum_{k=1}^{K} \sum_{i=1}^{n} \ln(\varphi(x_i|\theta_k))
\]

The last simplification of the sums of products is to be made clear in Exercise 4.1. Therefore,

\[
Q(\theta, \theta^o) = \sum_{k=1}^{K} \sum_{i=1}^{n} \ln(\alpha_k \varphi(x_i|\theta_k)) p(k|x_i, \theta^o)
\]

Both the terms in the latter expression can be maximized independently. The first term is maximized with respect to \(\alpha_k; k = 1, \ldots, K\), under the obvious condition \(\sum_{k=1}^{K} \alpha_k = 1\):

\[
\hat{\alpha}_{k,o} = \frac{1}{n} \sum_{i=1}^{n} p(k|x_i, \theta^o)
\]
The maximization of the second term results in the weighed mean and variance for each Gaussian component. Therefore, the EM algorithm for the Gaussian mixture identification is as follows:

**Initialization** Initial guesses \( \theta^{[0]} = [\hat{\alpha}_{k,[0]}, \hat{\mu}_{k,[0]}, \hat{\sigma}^2_{k,[0]} : k = 1, \ldots, K] \) for the MLEs.

**Iterative EM updating of the MLEs** At each iteration \( t = 1, 2, \ldots \),

**E-step:** Compute the responsibilities of each component \( k \) for the measurements \( x_i \in \mathbf{x} \):
\[
p(k|x_i, \theta^{[t]}) = \frac{\hat{\alpha}_{k,[t-1]} \varphi(x_i|\hat{\mu}_{k,[t-1]}, \hat{\sigma}^2_{k,[t-1]})}{\sum_{j=1}^{K} \hat{\alpha}_{j,[t-1]} \varphi(x_i|\hat{\mu}_{j,[t-1]}, \hat{\sigma}^2_{j,[t-1]})}
\]

**M-step:** Compute the updated weighed MLEs: \( \forall k = 1, \ldots, K \),
\[
\hat{\alpha}_{k,[t]} = \frac{1}{n} \sum_{i=1}^{n} p(k|x_i, \theta^{[t]}) \\
\hat{\mu}_{k,[t]} = \frac{1}{\hat{\alpha}_{k,[t]}} \sum_{i=1}^{n} x_i p(k|x_i, \theta^{[t]}) \\
\hat{\sigma}^2_{k,[t]} = \frac{1}{\hat{\alpha}_{k,[t]}} \sum_{i=1}^{n} (x_i - \hat{\mu}_{k,[t]})^2 p(k|x_i, \theta^{[t]})
\]

**Stopping rule:** Terminate the EM iterations when all the individual estimates converge to (almost) constant values representing the final MLEs.

In the mixture identification, the latent data are the unobserved membership indicators. In other problems, the latent data may be actual characteristics that should be observed but are missed.

### 4.2 General EM framework

In the general case, the observed data set, \( \mathbf{x} \), has the log-likelihood \( \ell(\theta; \mathbf{x}) \) depending on parameters \( \theta \) that should be estimated. Together with the latent or missed data set, \( \mathbf{x}^l \), the observations form the complete, or augmented data set, \( \mathbf{x}^{cm} = (\mathbf{x}, \mathbf{x}^l) \), with the log-likelihood \( \ell_0(\theta; \mathbf{x}^{cm}) \). This log-likelihood is based on the complete probability density function \( p(\mathbf{x}^{cm}; \theta) \). In the mixture identification case (Section 4.1), \( \mathbf{x}^l = \gamma \) and \( \ell_0(\theta; \mathbf{x}^{cm}) = \ell(\theta; \mathbf{x}, \gamma) \) of Eq. (4.2).
To avoid confusion, let $\theta'$ denote the “dummy” parameter $\theta$ in the complete log-likelihood $\ell_0(\theta' ; x^{cm})$. The EM algorithm starts with the initial guesses $\hat{\theta}^{[0]}$ for the parameters. At E-step of each iteration $t$, the conditional math expectation of the complete log-likelihood is computed as a function of the dummy parameter:

$$Q \left( \theta' ; \hat{\theta}^{[t-1]} \right) = \mathbb{E} \{ \ell_0(\theta' ; x^{cm}) | x, \hat{\theta}^{[t-1]} \} \tag{4.4}$$

For the mixture identification, this averaging is shown in Eq. (4.3). The first argument $\theta'$ represents the parameters that will be optimized in order to approach the desired MLEs. The second argument $\hat{\theta}^{[t-1]}$ relates to the parameters that are used to evaluate the math expectation. In other words, $x$ and $\hat{\theta}^{[t-1]}$ are constants at this step, $\theta'$ is a variable to be optimized later on, and $x^lt$ is a random variable governed by the probability distribution $p(x^lt | x, \hat{\theta}^{[t-1]})$. Therefore the math expectation in Eq. (4.4) can be represented as

$$\mathbb{E} \{ \ell_0(\theta' ; x^{cm}) | x, \hat{\theta}^{[t-1]} \} = \int_{x^{lt} \in \mathcal{S}} \ln \left( p(x, x^lt | \theta') \right) p(x^lt | x, \hat{\theta}^{[t-1]}) dx^lt \tag{4.5}$$

where $p(x^lt | x, \hat{\theta}^{[t-1]})$ is the conditional probability distribution of the latent data depending on both the observations $x$ and on the current parameter estimates, and $\mathcal{S}$ is the space of all the possible values of $x^lt$.

Then, at M-step the updated parameter estimate $\hat{\theta}^{[t]}$ is computed by maximizing the expectation $Q(\ldots)$ over $\theta'$:

$$\hat{\theta}^{[t]} = \arg \max_{\theta'} Q \left( \theta' ; \hat{\theta}^{[t-1]} \right)$$

Each iteration of the EM algorithm increases the log-likelihood, so that the whole iterative process converges to a local maximum of the likelihood function.

**Convergence of the EM process** To show it, let us start with the obvious relationships for the joint and conditional probabilities of the augmented $x^{cm} = (x, x^lt)$, observed $x$, and latent $x^lt$ data sets as well as for the relevant log-likelihoods:

$$\frac{\ln(p(x|\theta'))}{\ell(\theta' ; x)} = \ln \left( \frac{p(x, x^lt | \theta')}{p(x^lt | x, \theta')} \right) = \ln \left( \frac{p(x, x^lt | \theta')}{p(x^lt | x, \theta')} \right) - \ln \left( \frac{p(x^lt | x, \theta')}{\ell_1(\theta' ; x^lt | x)} \right)$$

$$\ln(p(x|\theta')) = \ln \left( \frac{p(x, x^lt | \theta')}{p(x | x^lt, \theta')} \right) = \ln \left( \frac{p(x, x^lt | \theta')}{p(x | x^lt, \theta')} \right) - \ln \left( \frac{p(x^lt | x, \theta')}{\ell_1(\theta' ; x^lt | x)} \right)$$

$$\ln(p(x|\theta')) = \ln \left( \frac{p(x, x^lt | \theta')}{p(x | x^lt, \theta')} \right) = \ln \left( \frac{p(x, x^lt | \theta')}{p(x | x^lt, \theta')} \right) - \ln \left( \frac{p(x^lt | x, \theta')}{\ell_1(\theta' ; x^lt | x)} \right)$$

Prof. A. A. Farag, Dr. G. Gimelfarb

Fall 2005
The conditional math expectation of Eqs. (4.4) and (4.5) applied to the log-likelihoods on the left and right sides of Eq. (4.6) yields

\[
\ell(\theta', x) = \mathbb{E}\{\ell_0(\theta', x^{|x|})|x, \hat{\theta}^{t-1}\} - \mathbb{E}\{\ell_1(\theta', x^{|x|})|x, \hat{\theta}^{t-1}\} \\
Q\left(\theta', \hat{\theta}^{t-1}\right) - \Psi\left(\theta', \hat{\theta}^{t-1}\right)
\]

because the left-hand log-likelihood does not depend on the latent data. The function \(\Psi(\ldots)\) is the expected log-likelihood of a density \(p(x^{|x|}|x, \theta')\) with respect to the same density but depending on \(\hat{\theta}^{t-1}\). Therefore, \(\Psi\left(\theta', \hat{\theta}^{t-1}\right) = \max_{\theta''} \Psi\left(\theta', \hat{\theta}^{t-1}\right)\). If
\[
\hat{\theta}^t = \arg\max_{\theta'} Q(\theta', \hat{\theta}^{t-1}),
\]
then \(\ell(\theta^t, x) - \ell(\hat{\theta}^{t-1}, x) \geq 0\), and the EM iteration never decreases the likelihood.

**Generalized EM**  A more general form of the M-step does not maximize \(Q(\theta', \hat{\theta}^{t-1})\) but finds instead \(\hat{\theta}^t\) such that \(Q(\hat{\theta}^t, \hat{\theta}^{t-1}) > Q(\hat{\theta}^{t-1}, \hat{\theta}^{t-1})\). In this case the EM algorithm is called generalized EM (GEM).

In practice, the term EM is frequently applied to any iterative process that increases at each step the likelihood of certain data, although this process may differ from the “true” abovementioned EM algorithms.

### 4.3 Hidden Markov models

Hidden Markov models (HMM) are typically used for 1D processes evolving in time or space (e.g. along a line). Let \(y_T = (y(1), \ldots, y(T))\) be a sequence of length \(T\) of discrete states \(y(t) \in \mathbb{Y} = \{y_1, \ldots, y_J\}\) such that the process can revisit a state at different steps \(t = 1, \ldots, T\), and not every state need be visited. The sequences \(y\) are modeled with a random first-order Markov chain of states produced with step-independent transition probabilities \(\alpha = \{a(y_i, y_j) \equiv p(y(t+1) = y_j | y(t) = y_i) : (y_i, y_j) \in \mathbb{Y}^2\}\) of being in state \(y_j\) at step \(t+1\) given the state \(y_i\) at step \(t\). The probability of each particular sequence is \(p(y|\alpha) = \prod_{t=1}^{T} a(y(t-1), y(t))\) where the probabilities \(a(y(0), y)\) of transitions from a

---

2 Generally, the transition probabilities may be asymmetric, i.e. \(a_{ij} \neq a_{ji}\), and the same state may appear successively, i.e. \(a_{ii} > 0\).
dummy initial state \( y(0) \) to all the states \( y \in \mathcal{Y} \) in order to produce an arbitrary random sequence are assumed to be known.

Let \( \mathbf{x}_T = (x(1), \ldots, x(T)) \) be a sequence of random discrete scalar observations\(^3\), \( x(t) \in \mathcal{X} = \{x_1, \ldots, x_K\} \). For any state \( y_i \) at step \( t \), there is a set of step-independent probabilities of each observed value \( x_k \) at the same step: \( \beta = \{b_{jk} = p(x(t) = x_k | y(t) = y_j) : x_k \in \mathcal{X}; y_j \in \mathcal{Y}\} \). Such a model of the observed values depending on the unobserved Markov-chain states is called a hidden Markov model. The central problems in HMM models are evaluation, decoding, and learning:

**Evaluation:** Given sets \( \alpha \) and \( \beta \) of transition and “observation-related” probabilities, determine the probability of a particular sequence \( \mathbf{x}_T \) of observations.

**Decoding:** Given observations \( \mathbf{x}_T \), find the most likely sequence of hidden states \( \mathbf{y} \in \mathcal{Y}^T \) that could produce these observations.

**Learning:** Given only a rough model “structure”, i.e. the number of states and the number of observed values, and a training set \( \mathcal{Z} \) of states and observations, estimate the probabilities \( \alpha \) and \( \beta \).

### 4.3.1 Evaluation of an HMM

The probability that the model produces a sequence \( \mathbf{x}_T^o \) of observations is

\[
p(\mathbf{x}_T^o) = \sum_{\mathbf{y}_T \in \mathcal{Y}^T} p(\mathbf{x}_T^o, \mathbf{y}_T) = \sum_{\mathbf{y}_T \in \mathcal{Y}^T} \prod_{t=1}^T p(x^o(t) | y(t)) p(y(t) | y(t-1))
\]

due to Markovianity of the state sequences. Such evaluation is computationally too complex: \( O(J^T T) \) computations. But it can be performed much faster in a recursive manner due to each term \( p(x^o(t) | y(t)) p(y(t) | y(t-1)) \) involves only \( x^o(t), y(t), \) and \( y(t-1) \).

**HMM Forward algorithm** Let \( \alpha_j(t) \) denote the probability that the HMM under consideration is in hidden state \( y_j \) at step \( t \) after generating first \( t \) observations \( x_1, \ldots, x_t \):

\[
\alpha_j(t) = \sum_{i=1}^J \alpha_i(t-1) a_{ij} b_{jk}(x_k = x^o(t)) \tag{4.7}
\]

Then the **Forward algorithm** is as follows:

\(^3\)The observations may be vector-valued as well.
Initialization: Given the probabilities $\alpha; \beta$, including the forward transitions from the dummy state $y(0) = y_0$ to each of the states $y(1) \in Y$, and given the observations $x_{ji}$, set $\alpha_i(0) = 1$ for $i = 1, \ldots, J$, and $t = 0$

Recursion: For $t \leftarrow t + 1$ until $t = T$, compute Eq. (4.7) for all $j = 1, \ldots, J$

Termination: Return $p(x^T_j) = \alpha(T + 1)$ where $\alpha(T + 1)$ denotes the probability of ending to the dummy final state $y(T + 1)$ absorbing the sequences ended in all the states at step $T$: $\alpha(T + 1) = \sum_{j=1,...,J} \alpha_j(T)$

Computational complexity of the Forward algorithm, $O(J^2T)$, is much less than of the straightforward exhaustion.

HMM Backward algorithm is the step-reversed Forward algorithm:

Initialization: Given the probabilities $\alpha; \beta$, including the backward transitions from the dummy state $y(T + 1)$ to each of the states $y(T) \in Y$, and given the observations $x_{ji}$, set $\beta_j(T + 1) = 1$ for $j = 1, \ldots, J$, and $t = T + 1$

Recursion: For $t \leftarrow t - 1$ until $t = 1$, compute for all $i = 1, \ldots, J$:

$$\beta_i(t) = \sum_{j=1}^{J} \beta_j(t + 1) a_{ij} b_{jk}(x_k = x^*(t + 1))$$

Termination: Return $p(x^T_i) = \beta(0)$ where $\beta(0)$ denotes the probability of ending to the dummy initial state $y(0)$ absorbing the sequences ended in all the states at step 1: $\beta(0) = \sum_{i=1,...,J} \beta_i(1)$

### 4.3.2 Decoding of an HMM

Decoding is actually a Bayesian MAP decision about a hidden sequence of states, given a sequence of observations:

$$y^* = \arg\max_{y \in Y^T} \prod_{t=1}^{T} p(y(t)|y(t-1))p(x^o(t)|y(t))$$

$$= \arg\max_{y \in Y^T} \sum_{t=1}^{T} \left( \log a_{ij}(y(t-1) = y_i) \wedge (y(t) = y_j) + \log b_{jk}(y(t) = y_j) \wedge (x^o(t) = x_k) \right)$$
This problem is solved by dynamic programming algorithm (DPA) of time complexity $O(J^2T)$ using memory of size $JT$ to keep locally optimum backward directions.

Let us denote, for simplicity, the log weight of transition at step $1$ as

$$g_1(y_j) = \log a_{ij} : (y(0)=y_0) \land (y(1)=y_j) + \log b_{jk} : (y(1)=y_j) \land (x(1)=x_k)$$

and at steps $t=2,\ldots,T$, as

$$g_t(y_i, y_j) = \log a_{ij} : (y(i-1)=y_i) \land (y(t)=y_j) + \log b_{jk} : (y(t)=y_j) \land (x(t)=x_k)$$

for all $i, j \in \mathbb{Y}$ and the known observation $x^\circ(t)$. The problem is to maximize the additive function

$$g(y^{[1]}, y^{[2]}, \ldots, y^{[T]}) = g_1(y^{[1]}) + g_2(y^{[1]}, y^{[2]}) + \ldots + g_T(y^{[T-1]}, y^{[T]}); \quad y^{[t]} \in \mathbb{Y}; \quad t=1,\ldots,T$$

At the initial step $t=1$ the DPA initializes the locally maximal function $G_1(y^{[1]}) = g_1(y^{[1]})$; $y^{[1]} \in \mathbb{Y}$, and the backward transition function $B_1(y^{[1]}) = y_0$ (the backward transition to the dummy state $y_0$ indicates the terminating condition). At each step $t=2,\ldots,T$, the DPA computes $G_t(\ldots)$ and builds one more backward transition function $B_t(\ldots)$ as follows: for every value $y^{[t]} \in \mathbb{Y},$

$$y^* = \arg\max_{y \in \mathbb{Y}} \{G_{t-1}(y) + g_t(y, y^{[t]})\} \rightarrow \begin{cases} B_t(y^{[t]}) = y^* \\ G_t(y^{[t]}) = G_{t-1}(y^*) + g_t(y^*, y^{[t]}) \end{cases}$$

The global maximum is $G_T(y^*_{[T]})$ where $y^*_{[T]} = \arg\max_{y \in \mathbb{Y}} G_T(y)$, and the corresponding sequence of states is restored in inverse order using the backward transition functions: for all $t=T,\ldots,2 y^*_{[t-1]} = B(y^*_{[t]})$.

### 4.3.3 HMM learning

The goal is to estimate the transition and observation probabilities $a$ and $b$ from a set of training sequences. It is unknown how to obtain Bayesian estimates or MLEs. But a reasonably good solution is given by the forward–backward algorithm being an instance of a GEM (generalized expectation-minimization).

In Eq. (4.7), $\alpha_j(t)$ was defined as the probability that the target sequence generated up to step $t$ is in state $y_j$. Similarly, let $\beta_j(t)$ be the probability that the remaining states
of the target sequence, i.e. from step $t + 1$ to step $T$, are generated starting from state $y_j$ at step $t$:

$$
\beta_j(t) = \begin{cases} 
0 & \text{if } (t = T) \land (y(T) \neq y_j) \\
1 & \text{if } (t = T) \land (y(T) = y_j) \\
\sum_{j=1}^{J} \beta_j(t+1) a_{ij} b_{jk} (x^{(t+1)} = x_k) & \text{otherwise}
\end{cases}
$$

This equation allows to determine probabilities of states in backward direction. But the true $\alpha$ and $\beta$ are unknown, so that $\alpha_j(t)$ and $\beta_j(t)$ are only their rough estimates.

Let $\gamma_{ij}(t)$ be the probability of transition from state $y_i(t-1)$ to $y_j(t)$ given that the entire training sequence $\mathcal{Z}_T$ is produced by any path:

$$
\gamma_{ij}(t) = \frac{\alpha_i(t-1) a_{ij} b_{jk} \beta_j(t)}{p(\mathcal{Z}_T|\theta)}
$$

where $p(\mathcal{Z}_T|\theta)$ is the probability that the training sequence is generated by any path.

The improved estimates of desired probabilities are as follows:

$$
\hat{\alpha}_{ij} = \frac{\sum_{t=1}^{T} \gamma_{ij}(t)}{\sum_{i=1}^{J} \sum_{c=1}^{K} \gamma_{ic}(t)} \quad \text{and} \quad \hat{b}_{jk} = \frac{\sum_{t=1}^{T} b_{jk}(t)}{\sum_{i=1}^{J} \sum_{c=1}^{K} b_{ic}(t)} \quad (4.8)
$$

Thus, the Baum–Welch, or forward–backward GEM algorithm is as follows:

- **Initialization**: $a_{ij}$, $b_{jk}$; training sequence $\mathcal{Z}_T$; convergence threshold $\theta$; $s = 0$

- **Step** $s$: for $s \leftarrow s + 1$, compute using Eq. (4.8) $\hat{\alpha}_{s-1}$ and $\hat{\beta}_{s-1}$ from $\alpha_{s-1}$ and $\beta_{s-1}$ and assign $a_{ij,s} = \hat{a}_{ij,s-1}$ and $b_{jk,s} = \hat{b}_{jk,s-1}$ for all $i, j, k$.

- **Termination**: if $\max_{i,j,k} \{|a_{ij,s} - a_{ij,s-1}|, |b_{jk,s} - b_{jk,s-1}|\} < \theta$

### 4.4 Exercises

**Exercise 4.1.** Show explicitly that $\sum_{\gamma_1=1}^{K} \sum_{\gamma_2=1}^{K} \cdots \sum_{\gamma_n=1}^{K} \delta(\gamma_i - k) \prod_{j=1}^{n} p(\gamma_j|x_j, \theta^o) = p(k|x_i, \theta^o)$ in Eq. (4.3)

**Exercise 4.2.** Describe the EM algorithm that identifies a mixture

$$
p(x|\theta) = \sum_{k=1}^{K} \alpha_k \psi(x_i|\theta_k); \quad \sum_{k=1}^{K} \alpha_k = 1
$$
of arbitrary parametric density functions \( \psi(x|\theta) \) given a collection \( x = (x_1, \ldots, x_n) \) of measurements sampled from \( p(x|\theta) \). Relate this algorithm to the general EM framework in Section 4.2.

**Exercise 4.3.** Apply the algorithm from Exercise 4.2 to identification of a mixture of exponential functions \( \psi(x|\mu, \sigma) = \frac{1}{2\sigma} \exp \left( -\frac{|x-\mu|}{\sigma} \right) \). Describe its convergence in terms of the general framework, in particular, Eqs. (4.4)–(4.6), reduced to this particular mixture.

**Exercise 4.4.** Consider a simple HMM in Fig. 4.1 generating a Markov chain \( y_T = (y(1), \ldots, y(T)) \) of two states randomly selected from the set \( \mathcal{Y} = \{ y_1, y_2 \} \) with the transitional probabilities

\[
\begin{align*}
   a_{11} & \equiv p(y(t) = y_1|y(t-1) = y_1) = 0.9; \quad a_{12} = p(y(t) = y_2|y(t-1) = y_1) = 1 - a_{11} = 0.1 \\
   a_{22} & \equiv p(y(t) = y_2|y(t-1) = y_2) = 0.5; \quad a_{21} = p(y(t) = y_1|y(t-1) = y_2) = 1 - a_{22} = 0.5
\end{align*}
\]

![Diagram of HMM](image)

**Figure 4.1:** The 2-state HMM model in Exercise 4.4.

The initial state, \( y(1) \), is equiprobable, i.e. the transitional probabilities from the dummy state \( y(0) \) are \( a_{01} = a_{02} = 0.5 \). A sequence of observations \( x^*_T = (x(1), \ldots, x(T)) \) with binary values \( X = \{ x_1 = 0, x_2 = 1 \} \) is produced by \( y_T \) with conditional probabilities

\[
\begin{align*}
   b_{11} & \equiv p(x(t) = x_1|y(t) = y_1) = 0.9; \quad b_{12} = p(x(t) = x_2|y(t) = y_1) = 1 - b_{11} = 0.1 \\
   b_{22} & \equiv p(x(t) = x_2|y(t) = y_2) = 0.9; \quad b_{21} = p(x(t) = x_1|y(t) = y_2) = 1 - b_{22} = 0.1
\end{align*}
\]

Let \( T = 10 \) and let \( x^c_{10} = (0, 0, 0, 1, 1, 0, 1, 0, 1) \). Show step-by-step how the hidden sequence of states is decoded by the DPA from Section 4.3.2.